

Introduction to Cloud Computing

What is Cloud?

Introduction to Cloud Computing

In simple words, Cloud computing is – Placing your data on someone else's datacenter, letting them manage underline hardware Infrastructure (optionally underline Database or applications too); while having your full control on the data, and accessing that data through Internet or dedicated network.

Cloud computing is a model for enabling **universal, on-demand** access to a **shared pool** of configurable computing resources (e.g., computer networks, servers, storage, applications and services), which can be **rapidly provisioned** and **released** with **minimal management effort** on **Pay-per-use basis**.

Free available cloud Examples:

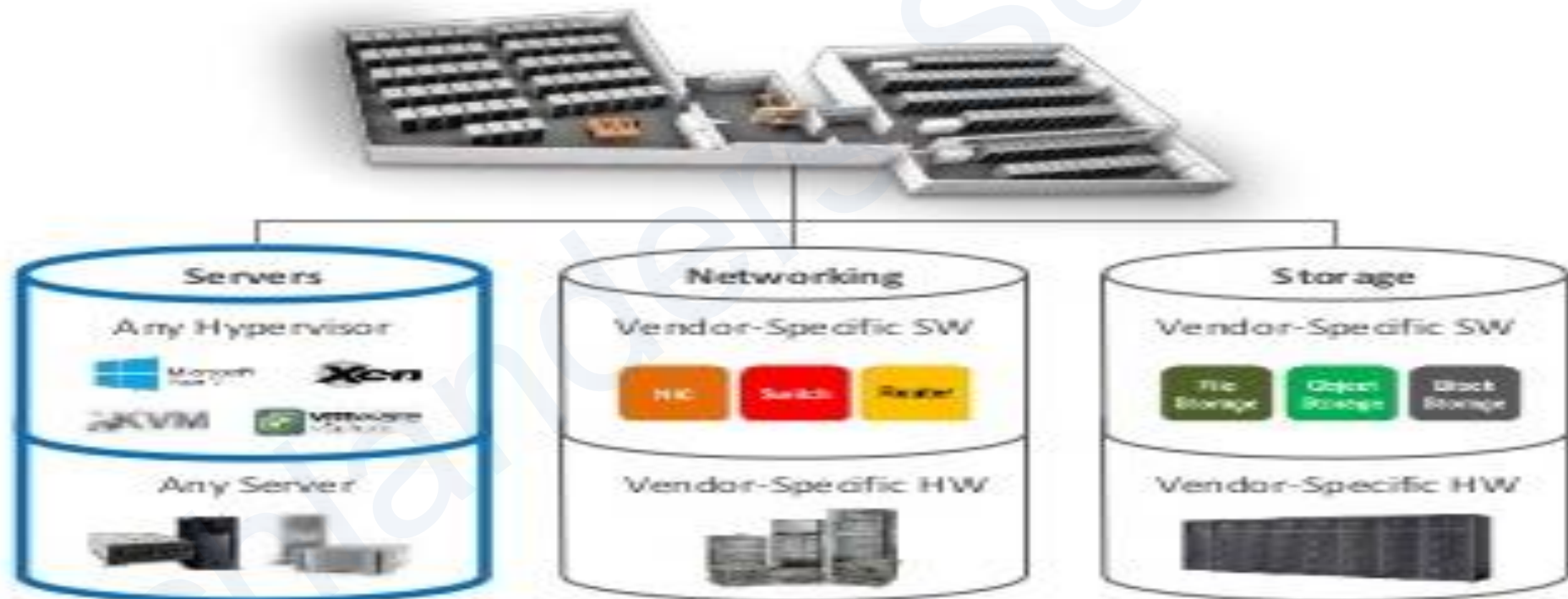
Paid available cloud Examples:

Gmail, IRCTC, WhatsApp/Facebook

AWS, Azure(Microsoft), Oracle Cloud


Traditional DataCenters

Traditional Data Center




Traditional DataCenters

Main issues with Traditional IT Infrastructure.

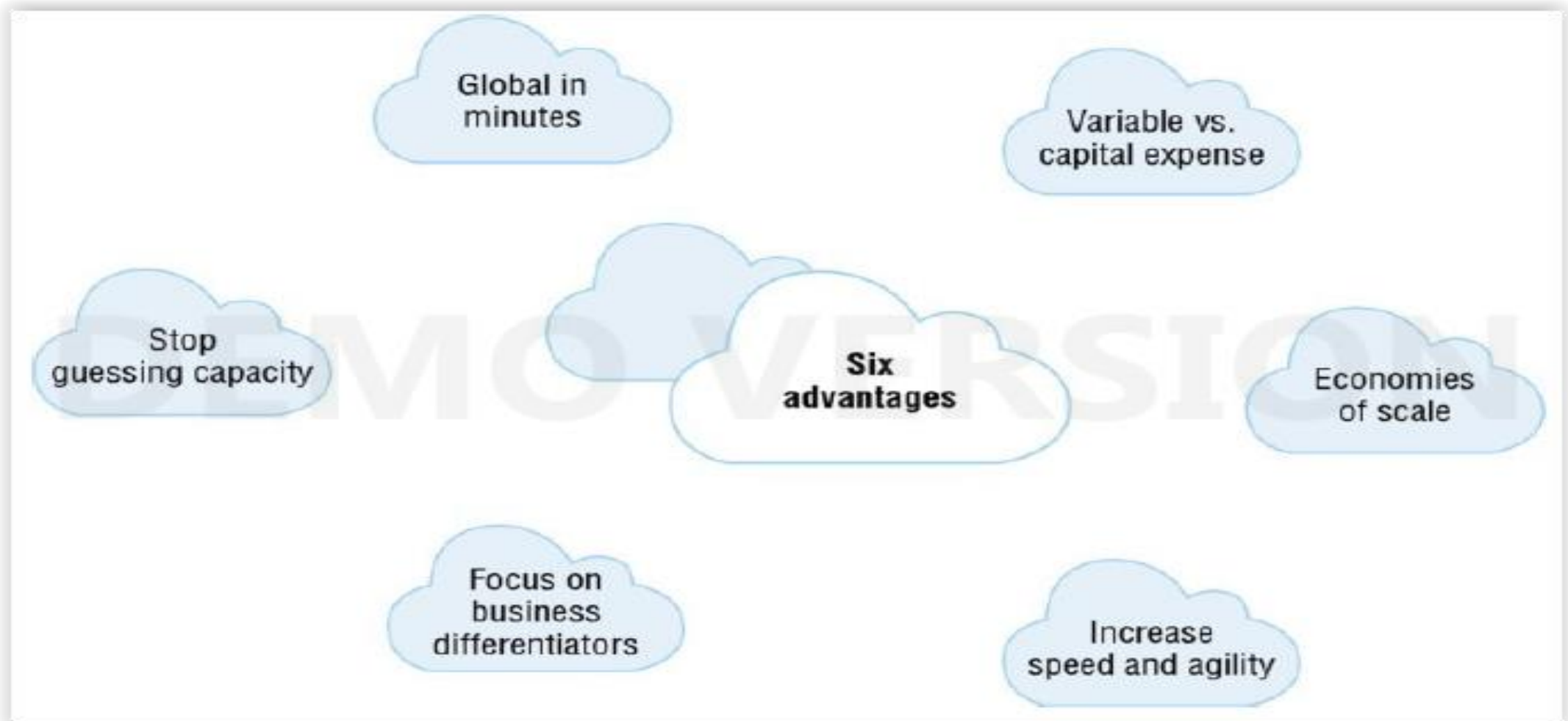
- Infrastructure is not a core business
 - Hard to Scale
 - Dedicated Infrastructure teams
 - Dedicated Datacenters
 - Dependency on vendors (servers, switches, cables etc.)
 - Underutilized Resources
 - High Cost
 - Difficult Capacity Planning
 - On-Spot demands were hard to manage
 - Provisioning resources was very time consuming
- 

Why cloud?

To overcome all of the discussed challenges, IT infrastructure domain drifted towards Service based model which is a real “cloud computing”

- No Dedicated Datacenter
 - No Different Infrastructure Teams
 - Higher/Faster Scalability
 - Elasticity
 - Pay per use model
 - Option to adopt high availability
 - Better performance
 - Instant provisioning
 - Optimized use of resources
 - On demand scaling to any extent
 - No to worry about capacity planning
- 

Cloud Advantages



Cloud Service Models

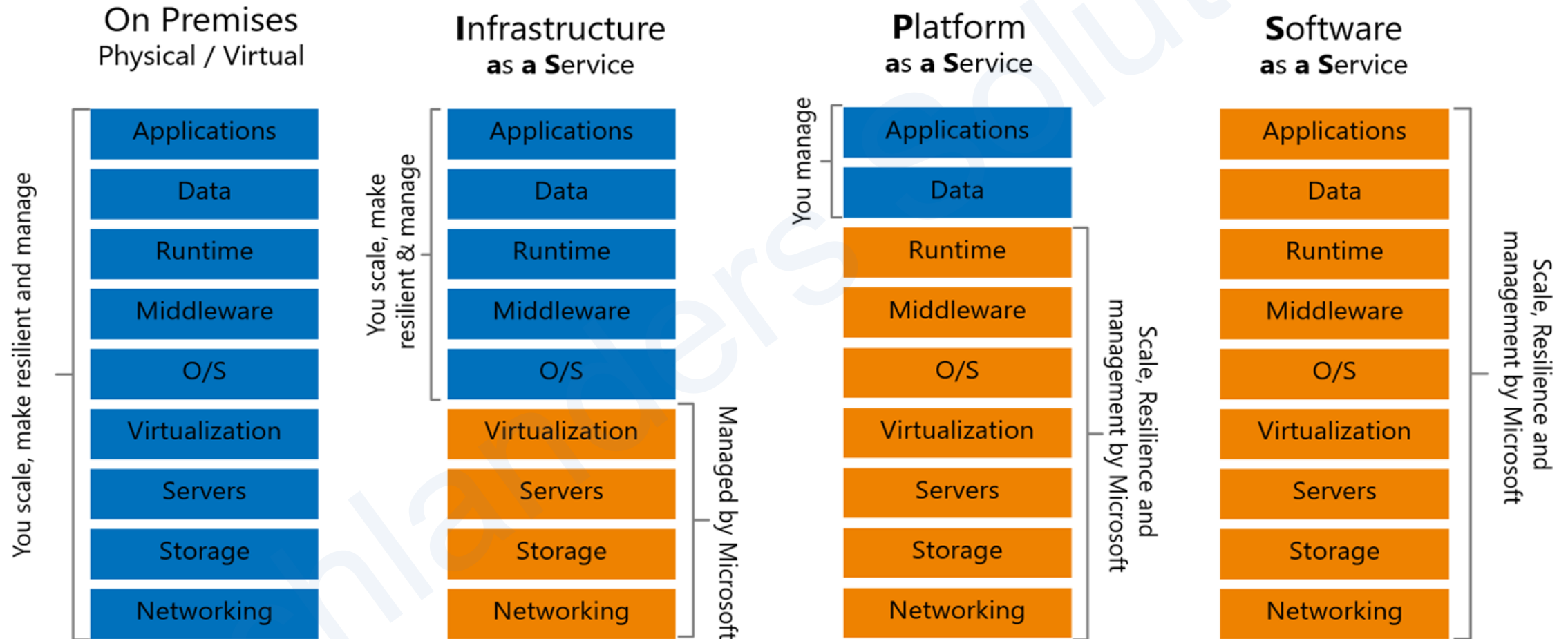
There are three Cloud Computing Service Models:

Infrastructure as a Service (IaaS)

Platform as a Service (PaaS)

Software as a Service (SaaS)

Responsibility- Who owns What?



Responsibility- Who owns What?

Pizza as a Service



Cloud Service Models - IaaS

IaaS is the most basic Cloud Service Model

It offers Underline Infrastructure for Compute, Storage and Networking

Infrastructure can be selected by customers as per their choice and Pay-per-use model.

Examples: Bare metal servers, virtual Instances, Load balancers



IaaS - Benefits

Drastic reduction in capital investment

Easily Scalable

Pay only for the used resources

High Flexibility

Reduced infrastructure support teams



Cloud Service Models - PaaS

Another service model, where cloud provider manages the OS & middleware part, along with IaaS

Provide capability to deploy applications on cloud infrastructure without managing underline Infra

Consumers are responsible for managing deployed applications and their environment specific configurations

Examples: web servers and databases



PaaS - Benefits

Includes all IaaS benefits

No upfront licensing cost

More reduction in Infrastructure support team

Rapid time to market



Cloud Service Models - SaaS

SaaS deliver complete application to the consumers over the internet.

Consumers are not responsible for managing any application or underlying infrastructure.

SaaS application are delivered as “one-to-many” model.

Examples: office365, Gmail, WhatsApp, JIRA, GIT, Service Now

SaaS - Benefits

Includes all discussed benefits which we get in PaaS

Ability to access from anywhere

Ability to access from multiple devices

No installations and maintenance requirements

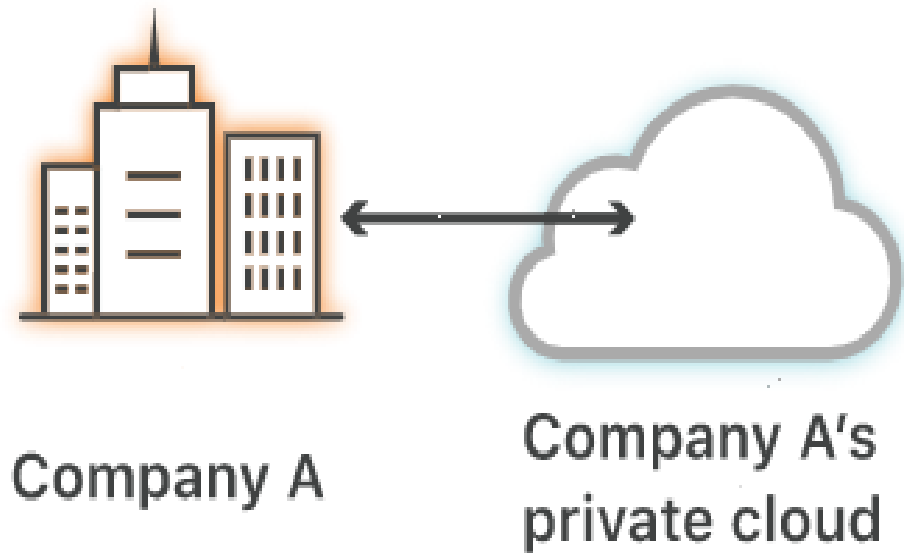
No Application management/Licensing Required



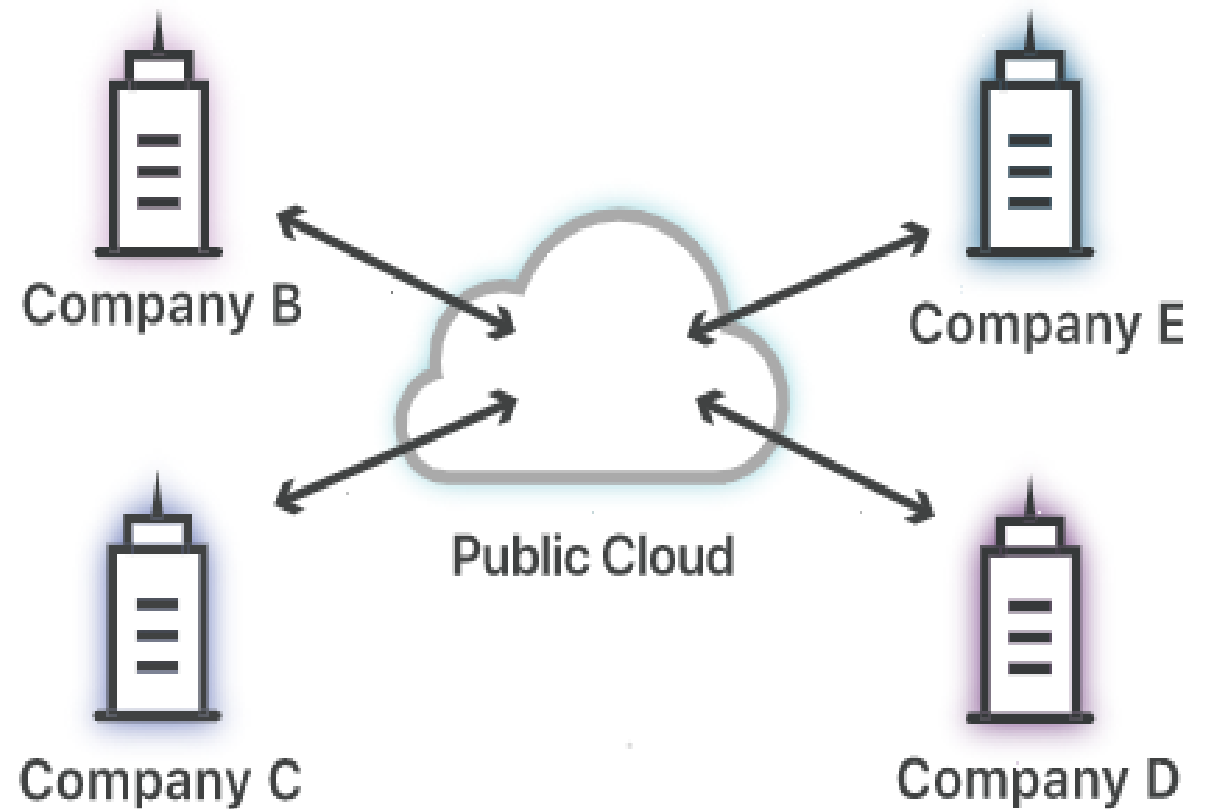
Cloud Essentials Characteristics



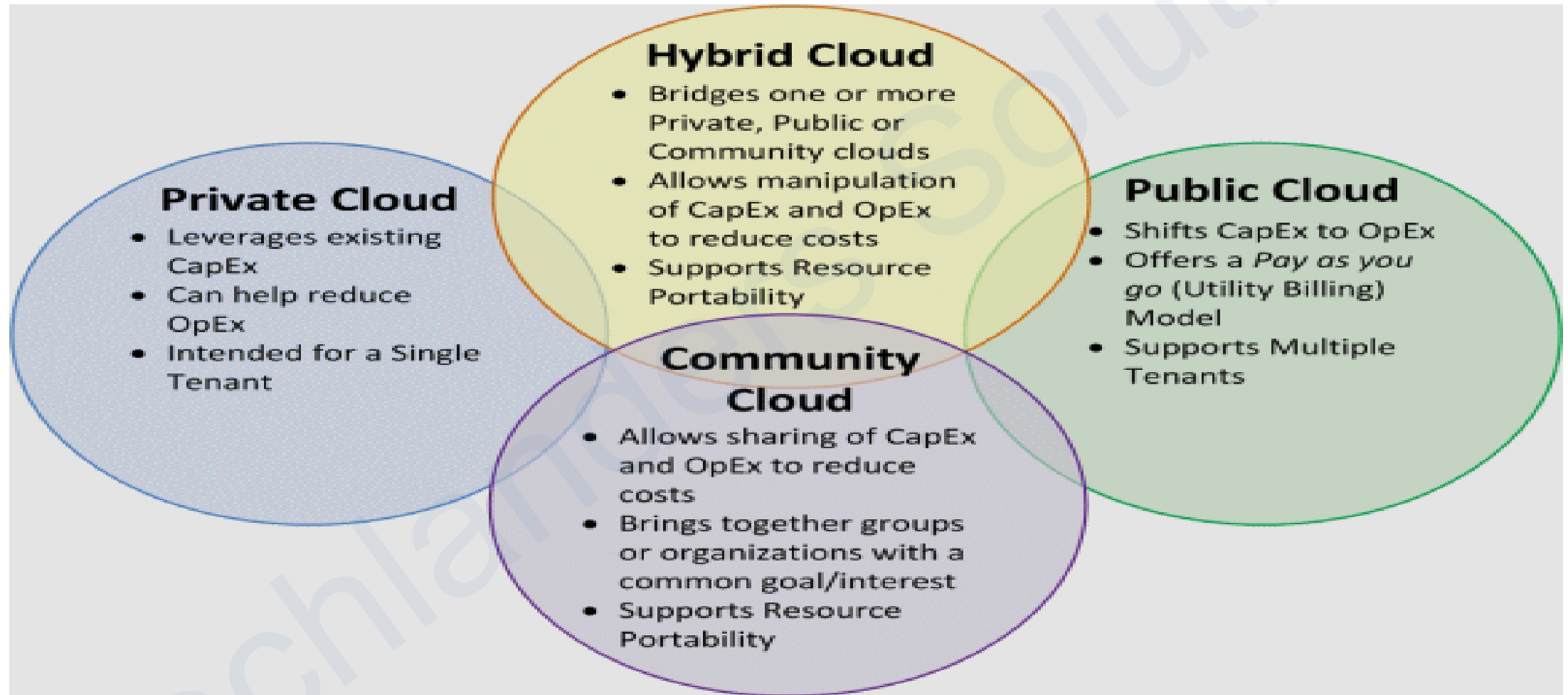
Private cloud



Public cloud shared by multiple companies



Cloud Deployments Types



Cloud's Major Use Cases



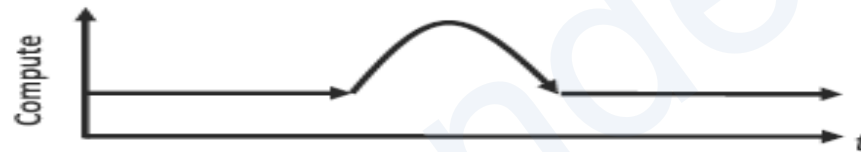
On and Off

On and off workloads (e.g. batch job)
Over provisioned capacity is wasted
Time to market can be cumbersome



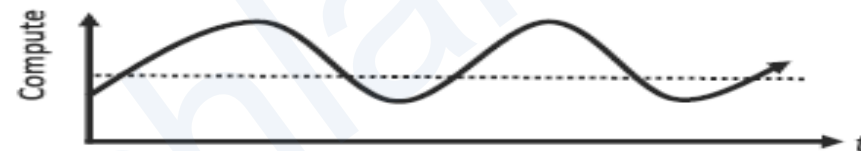
Growing Fast

Successful services need to grow/scale
Keeping up with growth is a big IT challenge
Cannot provision hardware fast enough



Unpredictable Bursting

Unexpected/unplanned peak in demand
Sudden spike impacts performance
Cannot over provision for extreme cases

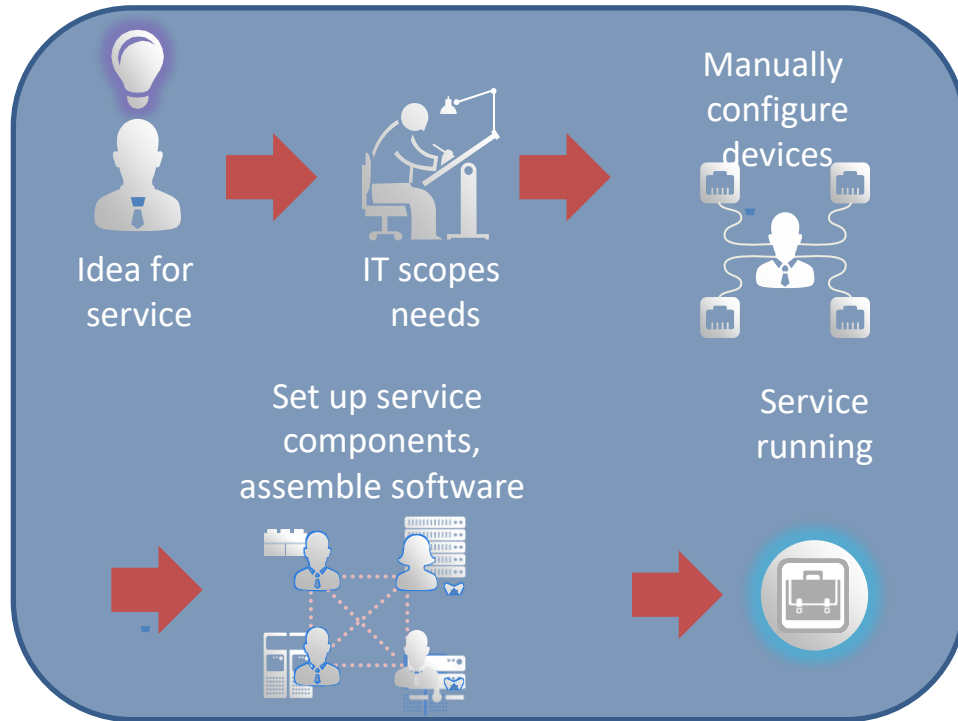


Predictable Bursting

Services with micro seasonality trends
Peaks due to periodic increased demand
IT complexity and wasted capacity

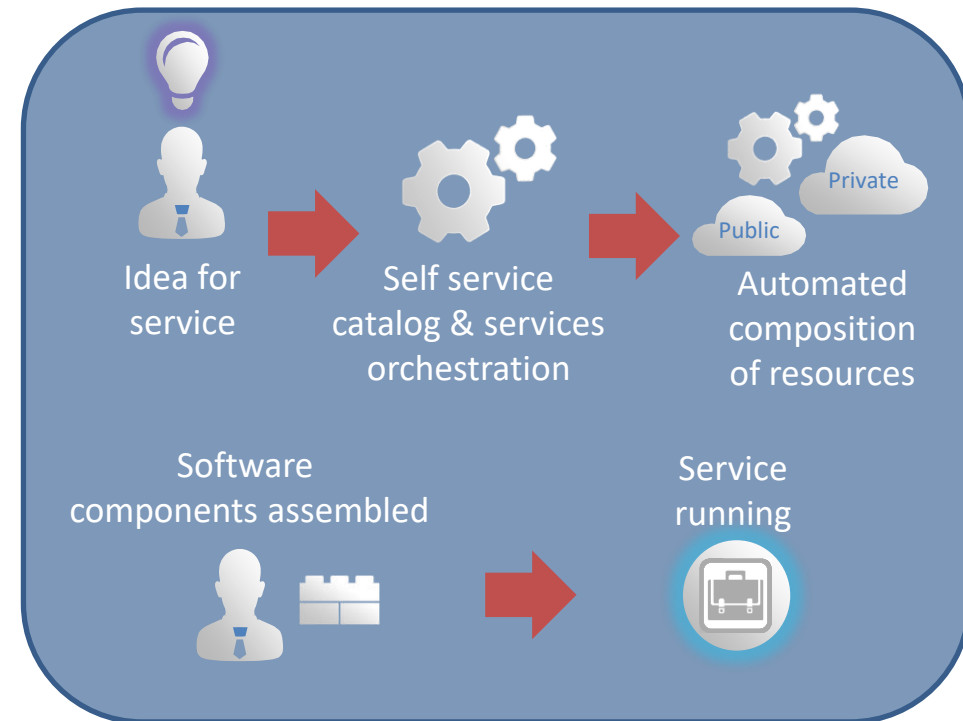
Business Impact of Cloud

Traditional Datacenter



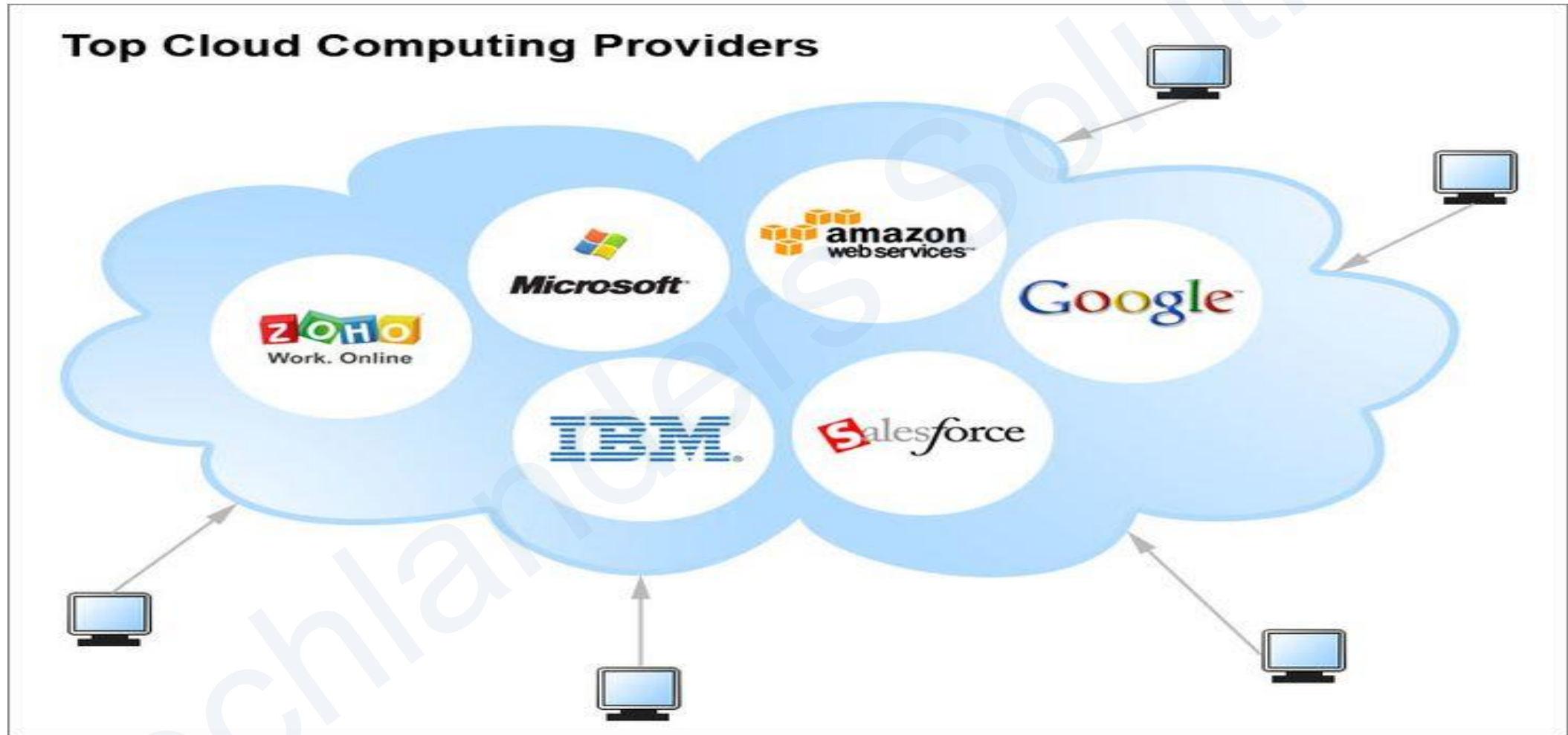
Time to Provision New Service: Months

Cloud Infrastructure

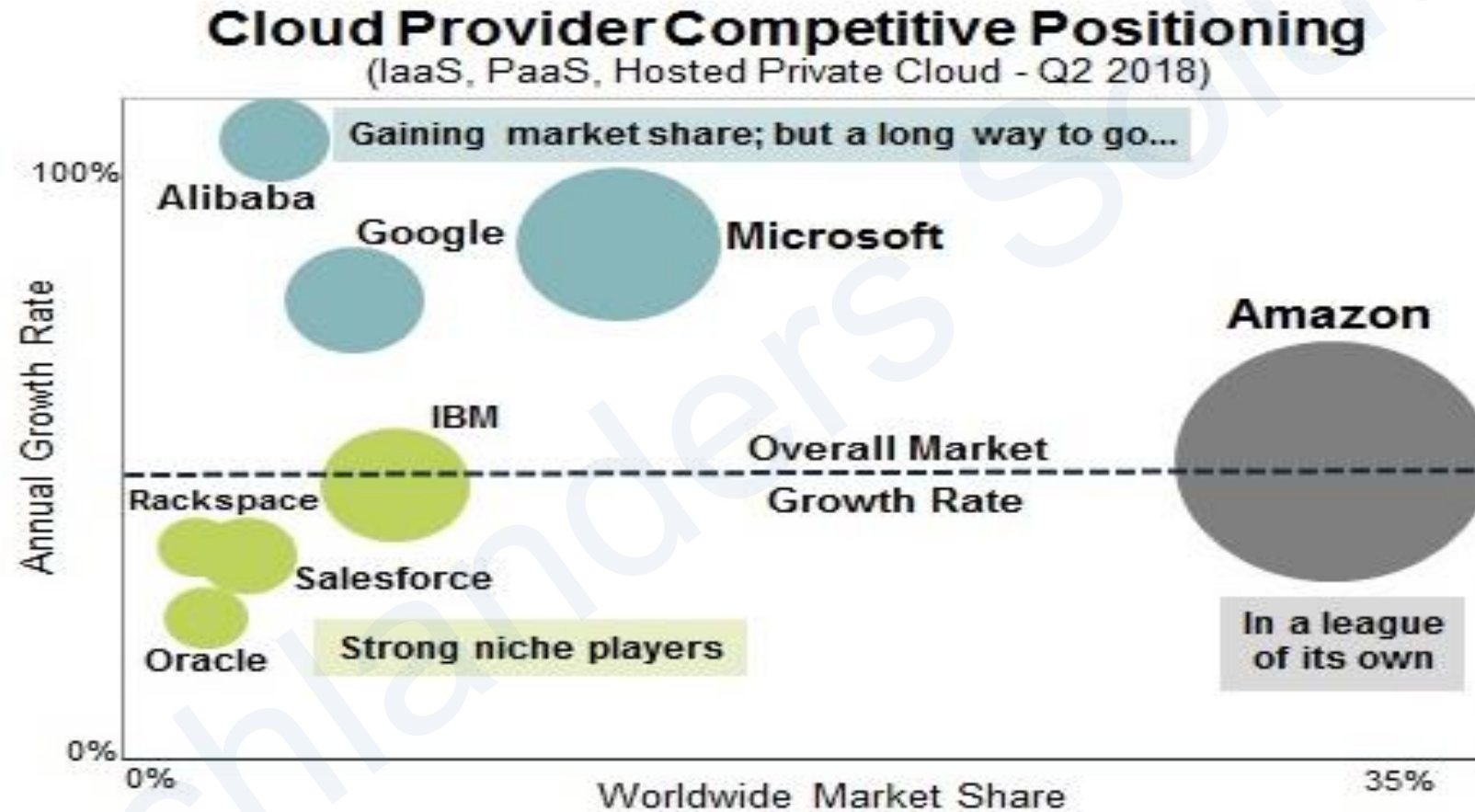


Time to Provision New Service: Minutes

Major Cloud Vendors

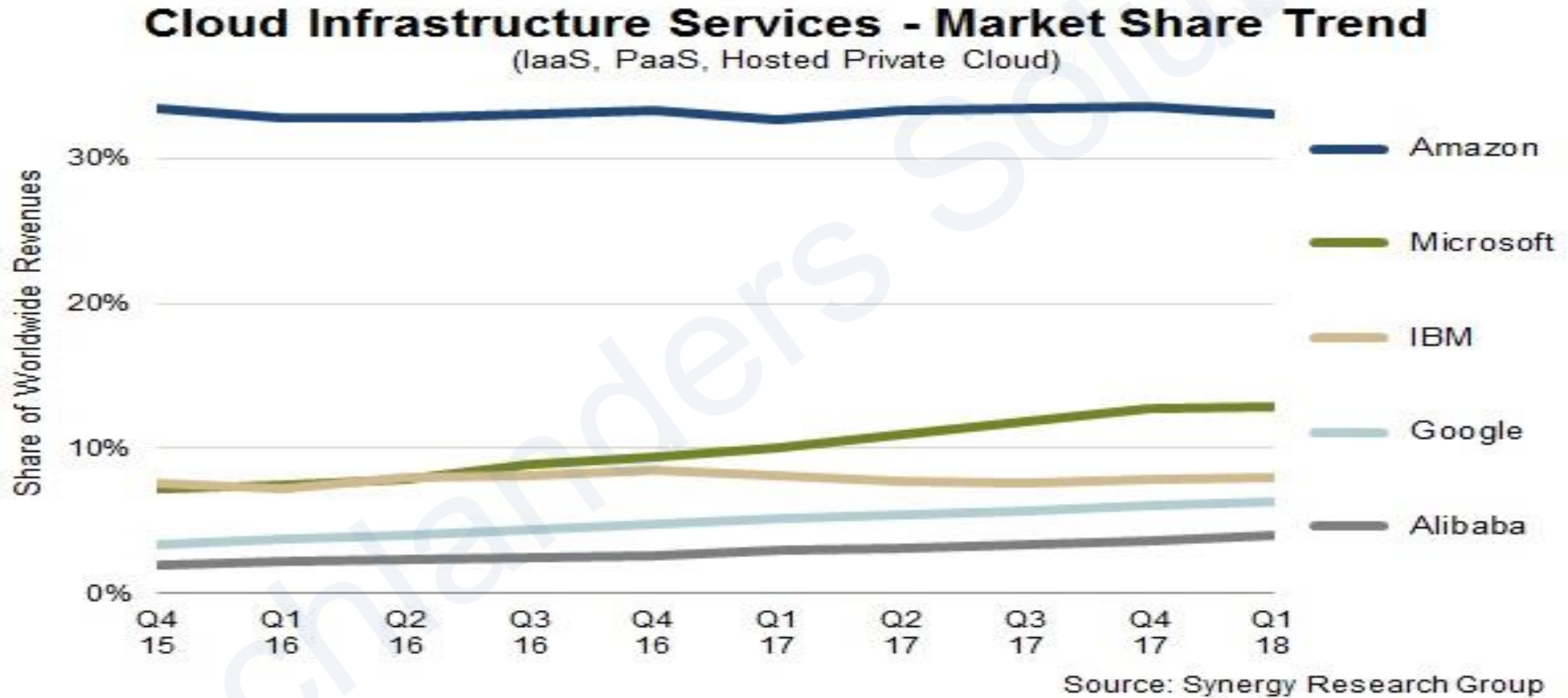


Who stands where?



Source: Synergy Research Group

Who stands where?



Knowledge Checks

- Which Service Level (IaaS, PaaS, SaaS) provides you most control?
- What is Hybrid Cloud?
- Can two public clouds be connected?
- Connecting two public clouds, will be know as public cloud or Hybrid?
- Cloud provided Database, is a PaaS or SaaS?

AWS (Amazon Cloud)

Amazon Web Services

AWS (Amazon Web Services) is a group of web services (also known as cloud services) being provided by Amazon since 2006.

AWS provides huge list of services starting from basic IT infrastructure like CPU, Storage as a service, to advance services like Database as a service, Serverless applications, IOT, Machine Learning services etc..

Hundreds of instances can be build and use in few minutes as and when required, which saves ample amount of hardware cost for any organizations and make them efficient to focus on their core business areas.

Currently AWS is present and providing cloud services in more than 190 countries.

Well-known for IaaS, but now growing fast in PaaS and SaaS.



Why AWS?

Low Cost: AWS offers, pay as you go pricing. AWS models are usually cheapest among other service providers in the market.


Instant Elasticity: You need 1 server or 1000's of servers, AWS has a massive infrastructure at backend to serve almost any kind of infrastructure demands, with pay for what you use policy.

Scalability: Facing some resource issues, no problem within seconds you can scale up the resources and improve your application performance. This cannot be compared with traditional IT datacenters.

Multiple OS's: Choice and use any supported Operating systems.

Multiple Storage Options: Choice of high I/O storage, low cost storage. All is available in AWS, use and pay what you want to use with almost any scalability.

Secure: AWS is PCI DSS Level1, ISO 27001, FISMA Moderate, HIPAA, SAS 70 Type II passed. In-fact systems based on AWS are usually more secure than in-house IT infrastructure systems.



AWS Global Infrastructure

AWS Regions:

- Geographic Locations
- Consists of at least two Availability Zones(AZs)
- All of the regions are completely independent of each other with separate Power Sources, Cooling and Internet connectivity.

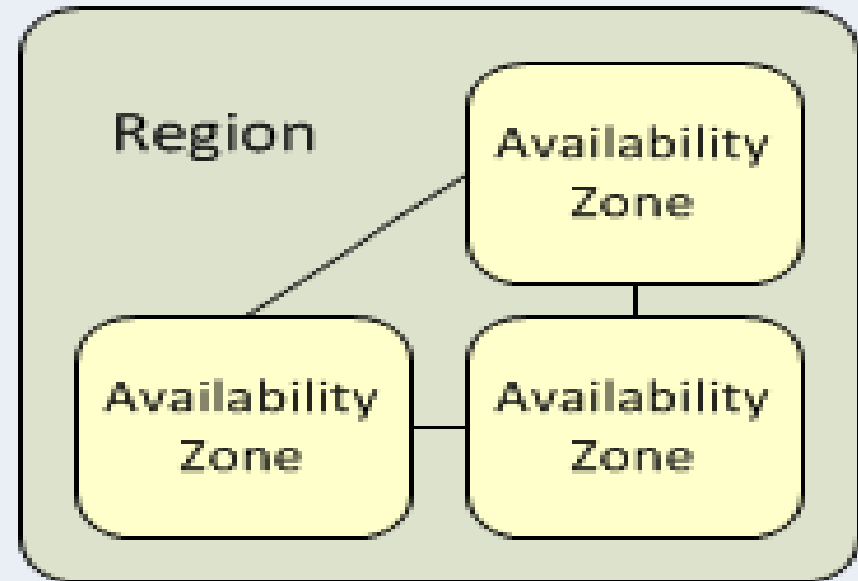
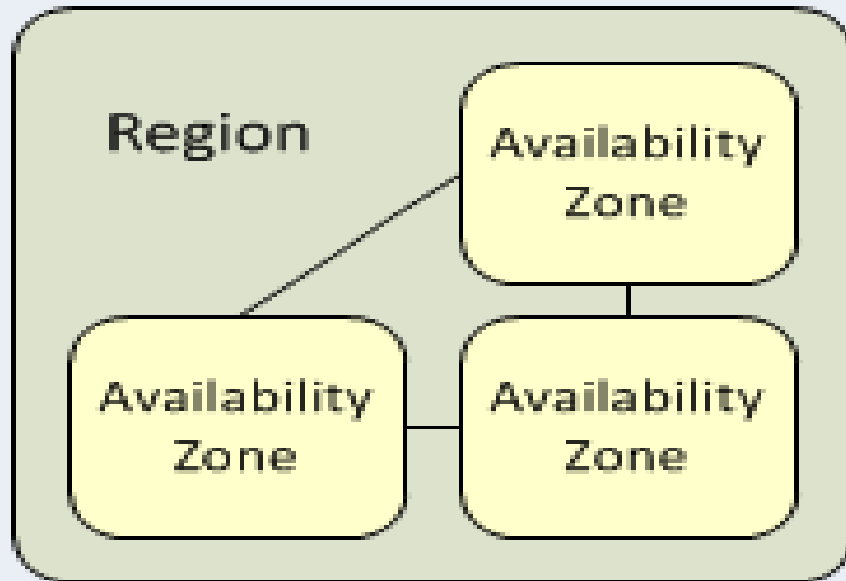
AWS Availability Zones

- AZ is a distinct location within a region
- Each zone is insulated (with low-latency links) from other to support single point of failures
- Each Region has minimum two AZ's
- Most of the services/resources are replicated across AZs for HA/DR purpose.

Note: Resources aren't replicated across regions unless you do so specifically.

AWS Global Infrastructure

Amazon Web Services



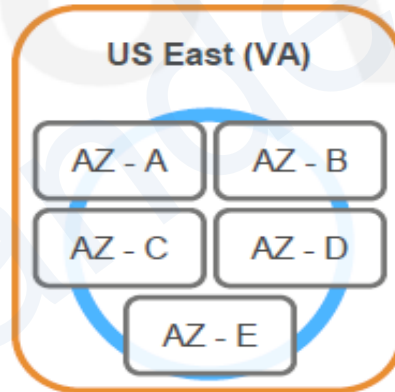
AWS Global Infrastructure

At least 2 AZs per region.

Examples:

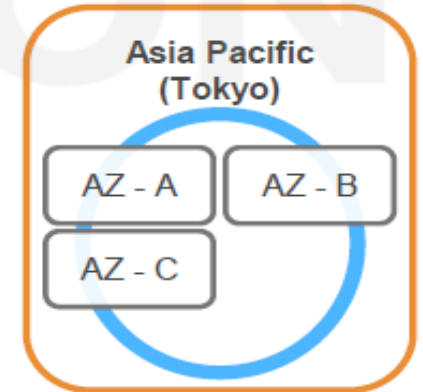
➤ US East (N. Virginia)

- us-east-1a
- us-east-1b
- us-east-1c
- us-east-1d
- us-east-1e



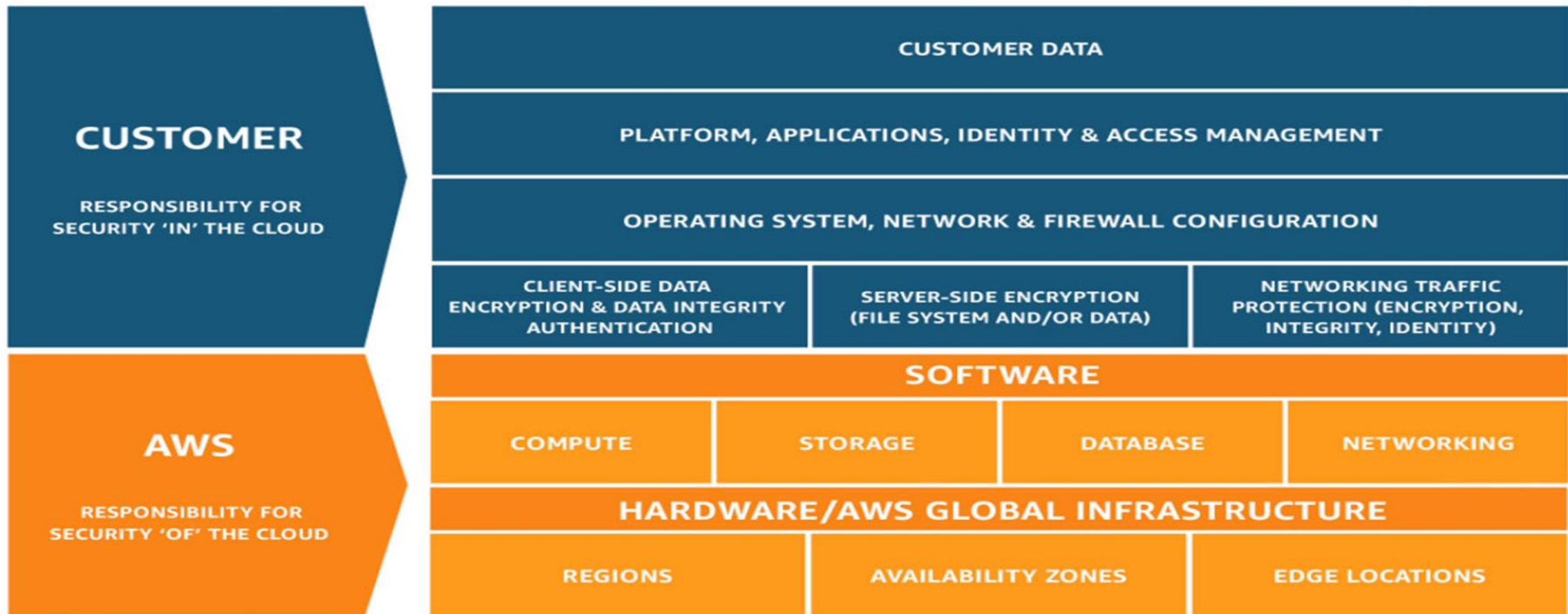
➤ Asia Pacific (Tokyo)

- ap-northeast-1a
- ap-northeast-1b
- ap-northeast-1c



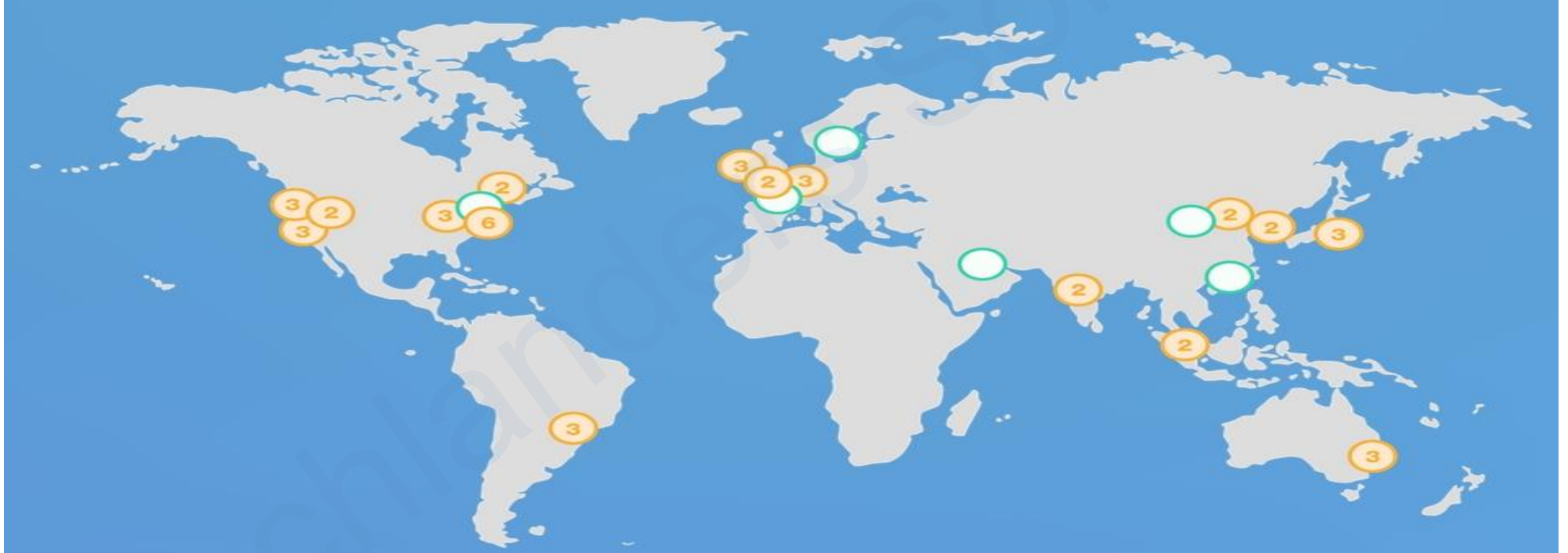
Note: Conceptual drawing only. The number of Availability Zones (AZ) may vary.

Shared Responsibility



AWS Global Infrastructure

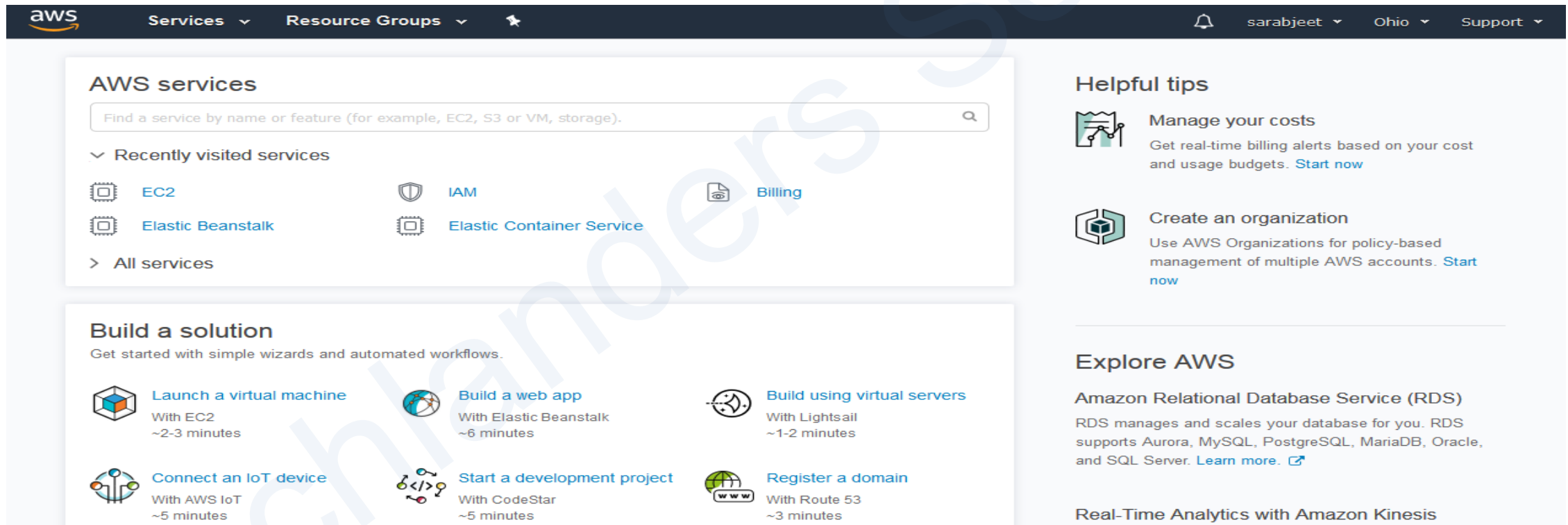
The AWS Cloud operates 44 Availability Zones within 16 geographic Regions around the world, with announced plans for 17 more Availability Zones and six more Regions in Bahrain, China, France, Hong Kong, Sweden, and a second AWS GovCloud Region in the US.



AWS Management Console

Simple and intuitive web-based user interface.

<https://console.aws.amazon.com/>



The screenshot displays the AWS Management Console interface. At the top is a dark navigation bar with the AWS logo, 'Services' and 'Resource Groups' dropdown menus, a star icon, and user information including a notification bell, the username 'sarabjeet', the region 'Ohio', and a 'Support' dropdown. Below the navigation bar, the main content area is divided into three sections. The 'AWS services' section features a search bar with the placeholder text 'Find a service by name or feature (for example, EC2, S3 or VM, storage)'. Below the search bar, under the heading 'Recently visited services', are links to 'EC2', 'Elastic Beanstalk', 'IAM', and 'Elastic Container Service', each accompanied by its respective icon. A link for 'Billing' is also present. An 'All services' link is at the bottom of this section. The 'Build a solution' section, titled 'Get started with simple wizards and automated workflows.', contains six cards: 'Launch a virtual machine' (With EC2, ~2-3 minutes), 'Build a web app' (With Elastic Beanstalk, ~6 minutes), 'Build using virtual servers' (With Lightsail, ~1-2 minutes), 'Connect an IoT device' (With AWS IoT, ~5 minutes), 'Start a development project' (With CodeStar, ~5 minutes), and 'Register a domain' (With Route 53, ~3 minutes). The right sidebar contains 'Helpful tips' with two items: 'Manage your costs' (Get real-time billing alerts based on your cost and usage budgets. Start now) and 'Create an organization' (Use AWS Organizations for policy-based management of multiple AWS accounts. Start now). Below this is the 'Explore AWS' section, which includes 'Amazon Relational Database Service (RDS)' (RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. Learn more.) and 'Real-Time Analytics with Amazon Kinesis'.

AWS services

Find a service by name or feature (for example, EC2, S3 or VM, storage).

Recently visited services

- EC2
- Elastic Beanstalk
- IAM
- Elastic Container Service
- Billing

> All services

Build a solution

Get started with simple wizards and automated workflows.

- Launch a virtual machine**
With EC2
~2-3 minutes
- Build a web app**
With Elastic Beanstalk
~6 minutes
- Build using virtual servers**
With Lightsail
~1-2 minutes
- Connect an IoT device**
With AWS IoT
~5 minutes
- Start a development project**
With CodeStar
~5 minutes
- Register a domain**
With Route 53
~3 minutes

Helpful tips

- Manage your costs**
Get real-time billing alerts based on your cost and usage budgets. [Start now](#)
- Create an organization**
Use AWS Organizations for policy-based management of multiple AWS accounts. [Start now](#)

Explore AWS

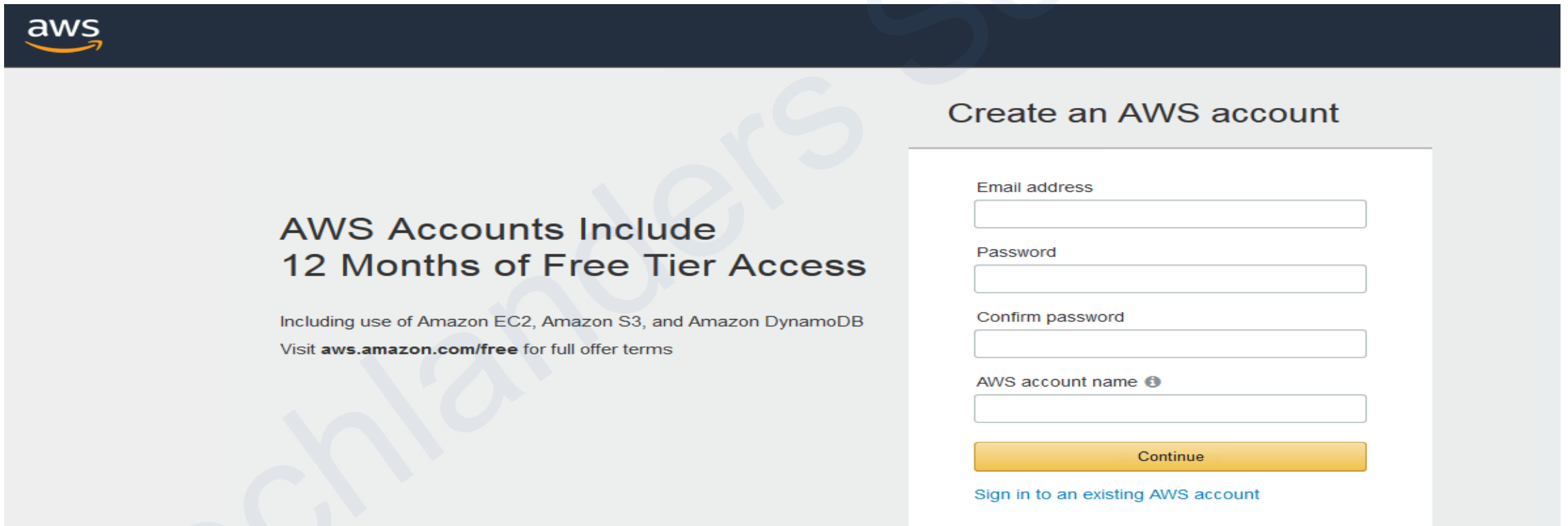
Amazon Relational Database Service (RDS)
RDS manages and scales your database for you. RDS supports Aurora, MySQL, PostgreSQL, MariaDB, Oracle, and SQL Server. [Learn more.](#)

Real-Time Analytics with Amazon Kinesis

LAB 1 : AWS Signup

Create a new account at

<https://portal.aws.amazon.com/billing/signup#/start>



The screenshot shows the AWS Signup page. On the left, there is a promotional message about the 12-month free tier. On the right, there is a form to create a new AWS account. The form includes fields for Email address, Password, Confirm password, and AWS account name, followed by a Continue button and a link to sign in to an existing account.

aws

Create an AWS account

**AWS Accounts Include
12 Months of Free Tier Access**

Including use of Amazon EC2, Amazon S3, and Amazon DynamoDB
Visit aws.amazon.com/free for full offer terms

Email address

Password

Confirm password

AWS account name ⓘ

Continue

[Sign in to an existing AWS account](#)

AWS SIGNUP

After creating the account

andhi Nagar

of suite, unit, building, floor, etc

Province or region

Code

Amazon Internet Services Pvt. Ltd. Customer

Customers with an India contact address are now required to register with Amazon Internet Service Private Ltd. (AISPL), the local seller for AWS infrastructure services in India.

Click here to indicate that you have read and agree to the terms of the [AISPL Customer Agreement](#)

Create Account and Continue

Payment Information

We use your payment information to verify your identity and only for usage in excess of the AWS Free Tier Limits. [We will not charge you for usage below the AWS Free Tier Limits.](#) For more information, see the [frequently asked questions](#).



As part of our card verification process we will charge INR 2 on your card when you click the "Secure Submit" button below. This will be refunded once your card has been validated. Your bank may take 3-5 business days to show the refund. Mastercard/Visa customers may be redirected to your bank website to authorize the charge.

Credit/Debit card number

Expiration date

10 ▼

2019 ▼

Cardholder's name

Select a Support Plan

AWS offers a selection of support plans to meet your needs. Choose the plan that best aligns with your AWS usage. [Learn more](#)



Basic Plan

Free

- Included with all accounts
- 24x7 self-service access to AWS resources
- For account and billing issues only
- Access to Personal Health Dashboard & Trusted Advisor



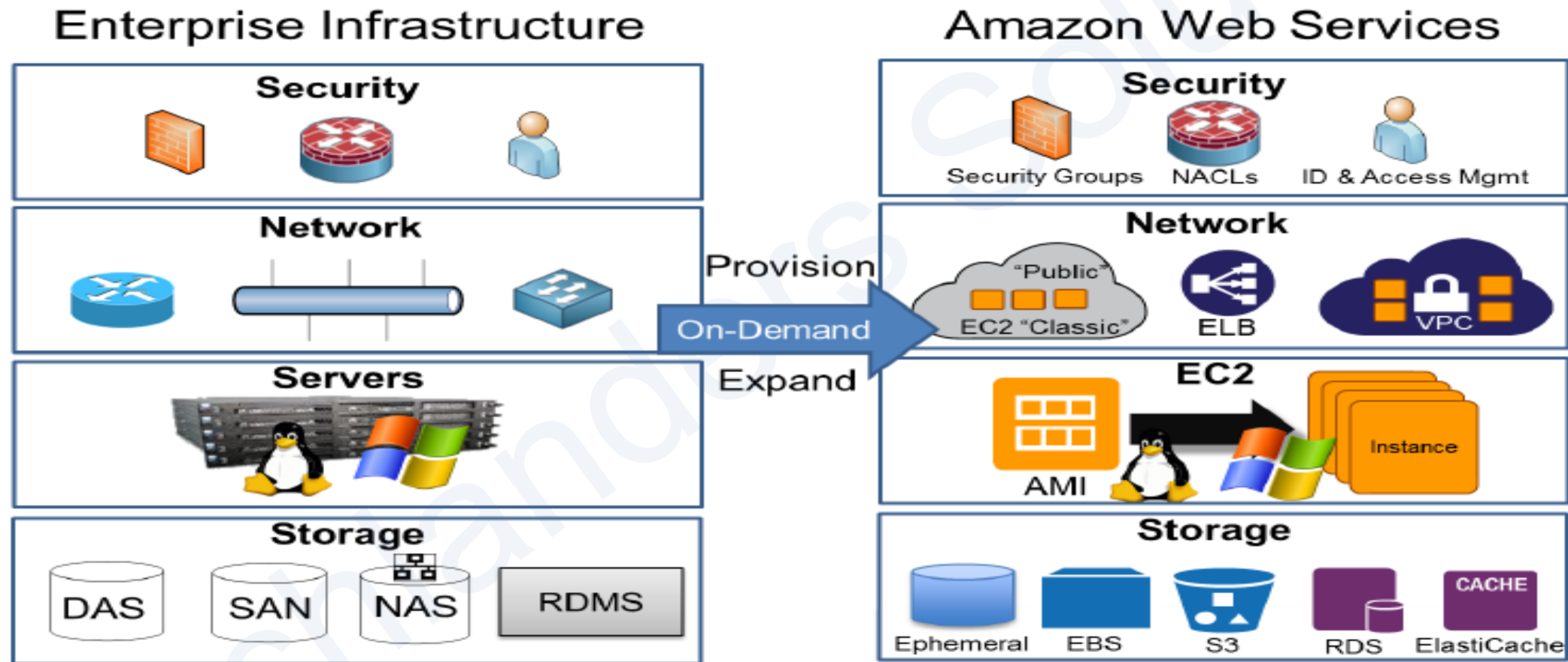
Developer Plan

From \$29/month

- For early adoption, testing and development
- Email access to AWS Support during business hours
- 1 primary contact can open an unlimited number of support cases
- 12-hour response time for nonproduction systems

Need Enterprise level support?

AWS Core Infrastructure Services



AWS Security

Physical Security:

24/7 trained security staff

AWS data centers in nondescript and undisclosed facilities

Two-factor authentication for authorized staff

Authorization for data center access

Multiple approval based change process



AWS Security

Hardware, Software, and Network :

Authentication and authorization in place

RBAC based access control mechanism

Firewall and other boundary devices

Security at Server level, Application level and Network level

AWS monitoring tools

Services to log AWS resources access



AWS Security



Amazon Resource Names (ARNs)

Amazon Resource Names (ARNs) uniquely identify AWS resources.

We require an ARN when you need to specify a resource unambiguously across all of AWS, such as in IAM policies, API calls etc.

ARN have a specific format:

arn:partition:service:region:account-id:resourcetype/resource

IAM user name

arn:aws:iam::123456789012:user/David


IAM instance id:

arn:aws:ec2:region:account-id:dedicated-host/host_id

Eg. arn:aws:ec2:us-east-1:123456789012:dedicated-host/h-12345678

AWS Access Credentials

AWS resources can be access using several authentication methods:

- IAM User-id / Password
 - Account ID/ AK (Access Key)/ SK (Security Key)
 - Certificates
 - Key pairs
- 

Resource Management Tools

AWS Management Console

AWS Console Mobile App (View resources)

AWS Command line interface

AWS Toolkit for PowerShell

AWS-Shell



AWS IAM

AWS Identity and Access Management

(IAM)

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources for your users. You use IAM to control who can use your AWS resources (***authentication***) and what resources they can use and in what ways (***authorization***).

Shared access to your AWS account

Granular permissions

Secure access to AWS resources for applications that run on Amazon EC2

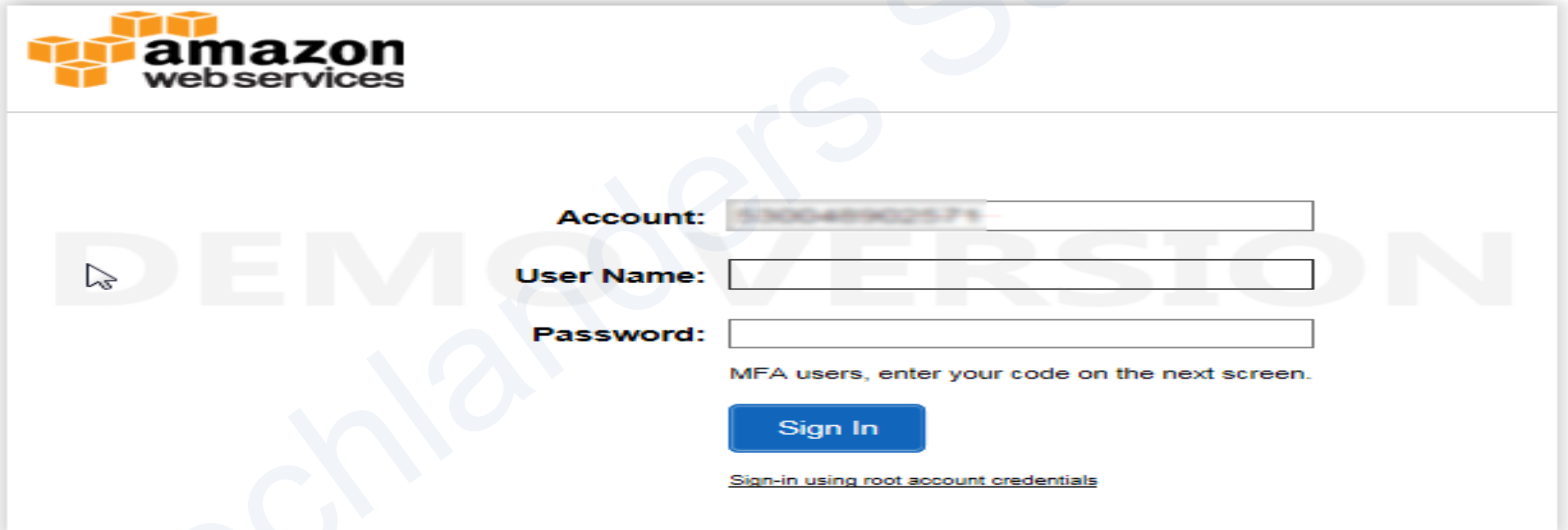
Integrated with many AWS services

Free to use



IAM - Users

You can create IAM users to distribute your work among team/users.
Different permissions can be given to IAM users



The screenshot shows the Amazon IAM console sign-in interface. At the top left is the Amazon Web Services logo. The main area contains three input fields: 'Account:' with a pre-filled value '111111111111', 'User Name:', and 'Password:'. Below these fields is a note for MFA users and a 'Sign In' button. A link for root account sign-in is at the bottom.

Account:

User Name:

Password:

MFA users, enter your code on the next screen.

[Sign In](#)

[Sign-in using root account credentials](#)

LAB 27 : IAM User

Create an IAM User

In this exercise, you will create an IAM user who can perform all administrative IAM functions. Then you will log in as that user so that you no longer need to use the root user login. Using the root user login only when explicitly required is a recommended security practice (along with adding MFA to your root user).


1. While logged in as the root user, create a new IAM user called user1.
2. Provide Access type -> Programmatic access and Management console access
3. Set a console password and click next
4. Logout from root and Use the customized sign-in link to sign in as Administrator
5. Try to login and check if you are able to access the s3 buckets in new user.
6. Were to able to see? What about other services?

IAM - Groups

Users are part of IAM groups

Common policies can be applied on groups

▼ Summary

Group ARN:	arn:aws:iam::242959498941:group/read_group 
Users (in this group):	2
Path:	/
Creation Time:	2018-05-02 22:21 UTC+0530

Users

Permissions

Access Advisor


This view shows all users in this group: **2 Users**

User	Actions
 gagandeep	Remove User from Group
 Tejas	Remove User from Group

LAB 26 : IAM Group

Create an IAM Group

In this exercise, you will create a group for all IAM administrator users and assign the proper permissions to the new group. This will allow you to avoid assigning policies directly to a user later in these exercises.

1. Log in as the root user.
 2. Go to **Security, Identity & Compliance** → **IAM** → **Groups**
 3. Create an IAM group provide name Elevated_Users
 4. Don't select any policy and click create.
 5. Add user created to the group
- 

AWS

Compute Services

Virtual Machines

What are the bare minimum infrastructure requirements for any application?

Hardware (RAM, Processor, Processor type, HBA's, etc.)

OS Disk & Data Disk

OS Images

Network

Security (Firewall, Networking, Access Mechanism)

Credentials



AWS Elastic Compute Cloud

Amazon **EC2** stands for **E**lastic **C**ompute **C**loud, and is the Primary AWS web service.

Provides Resizable compute capacity

Reduces the time required to obtain and boot new server instances to minutes

There are two key concepts to Launch instances in AWS:

- Instance Type**

- AMI**

EC2 Facts:

Scale capacity as your computing requirements change

Pay only for capacity that you actually use

Choose Linux or Windows OS as per need.

Deploy across AWS Regions and Availability Zones for reliability/HA

Only X86 based OS supported. So platform specific OS are not supported.

Instance Types (EC2)

The instance type defines the different virtual hardware Models (sizes) supporting an Amazon EC2 instance.

There are dozens of instance types available, varying in the following dimensions:

- Virtual CPUs (vCPUs)

- Memory

- Storage (size and type)

- Network performance

- Graphical Processing

Instance types are grouped into families based on the ratio of these values to each other.

Instance Types (EC2)

EC2 instance types are optimized for different use cases and come in multiple sizes. This allows you to optimally scale resources to your workload requirements.

AWS uses Intel® Xeon® processors for EC2 instances, providing customers with high performance and value.

Consider the following when choosing your instances: Core count, memory size, storage size and type, network performance, and CPU technologies.

Hurry Up and Go Idle - A larger compute instance can save you time and money, therefore paying more per hour for a shorter amount of time can be less expensive.

Instance Types with AWS

General
purpose



Compute
optimized



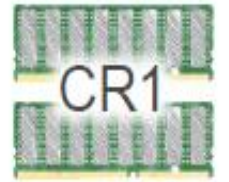
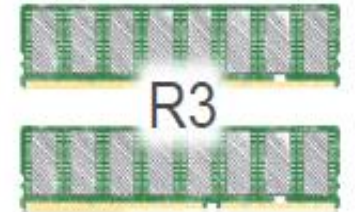
Storage and IO
optimized



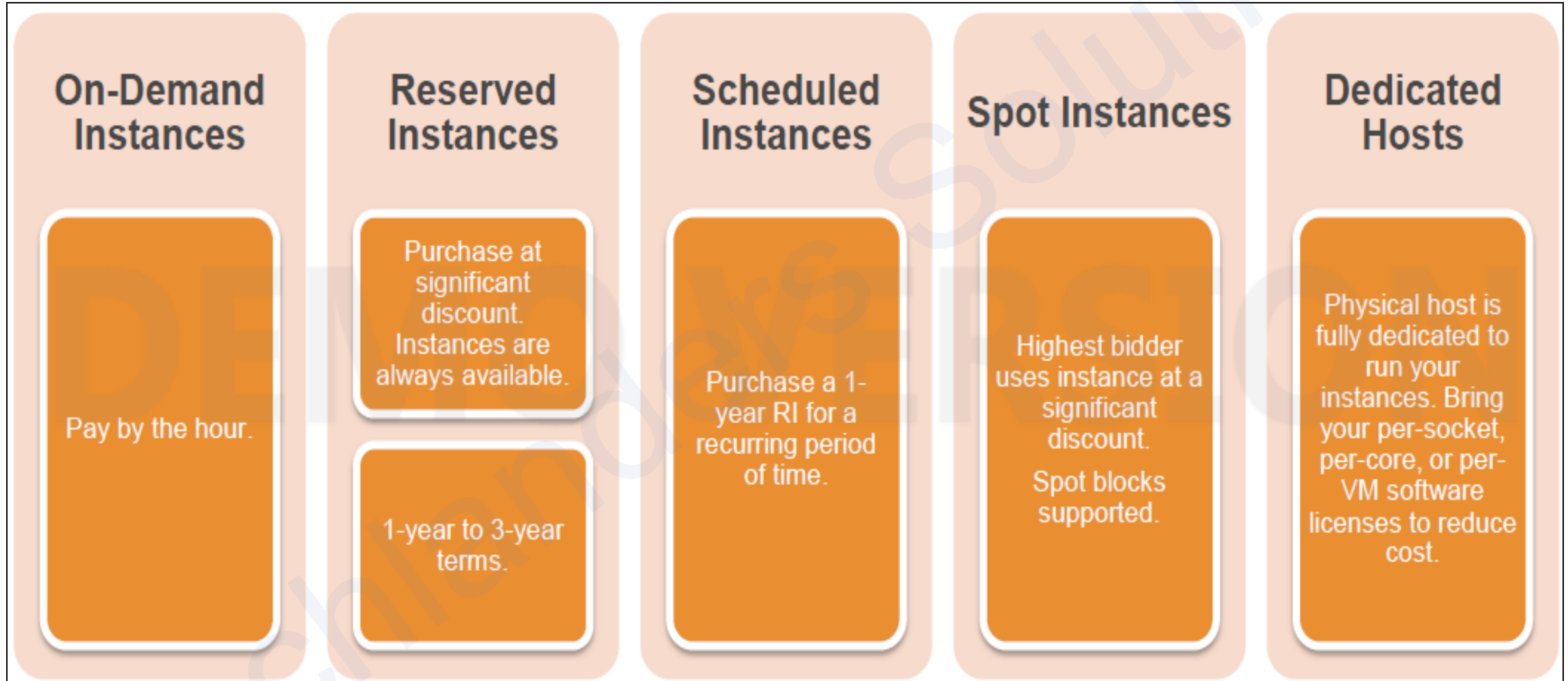
GPU
enabled



Memory
optimized



EC2 Pricing Models



AMI's

Stands for **Amazon Machine Image**

AMI is a template for the root volume for the instance (example: an OS image, a webserver, an application server etc.)

It also includes the launch permissions that control which AWS accounts can use the AMI to launch instances.

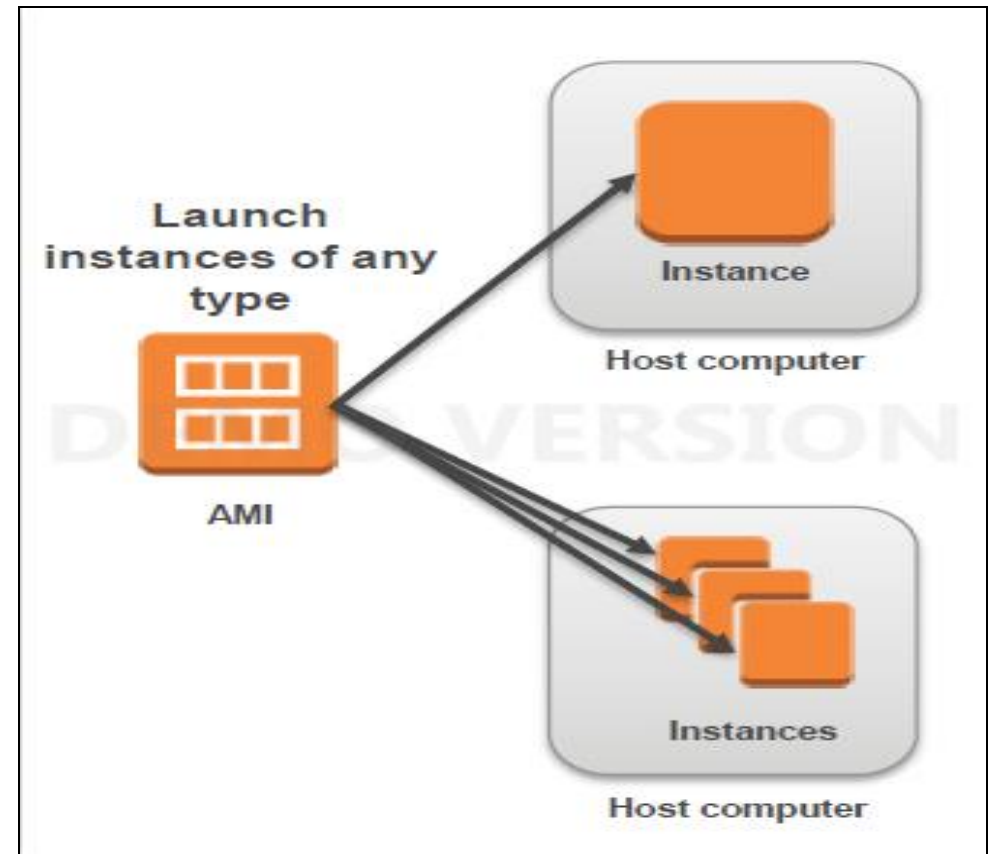
All AMIs are based on x86 OSs, either Linux or Windows.

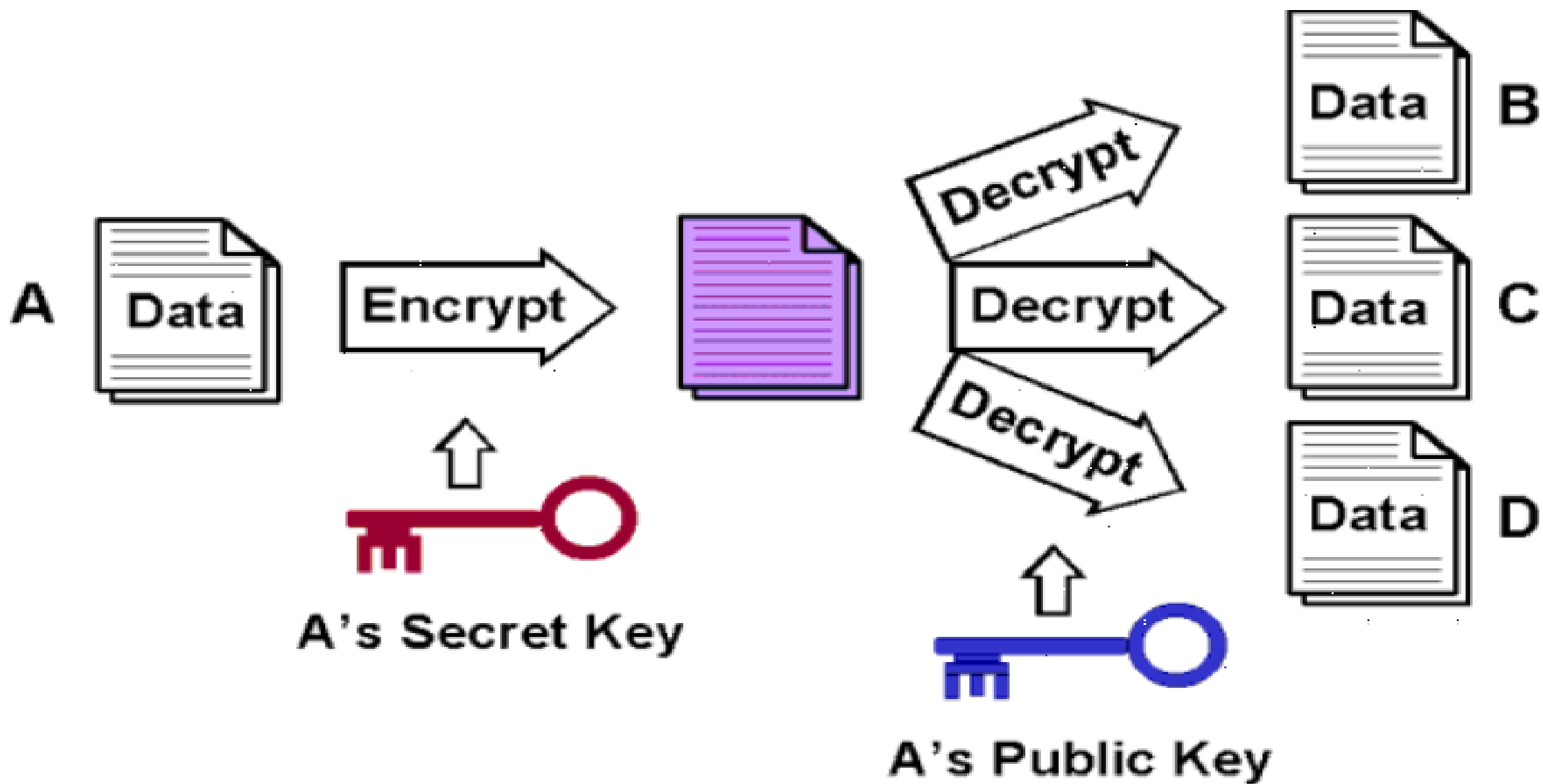


Instances and AMI's

Select an AMI based on:

- Region
- Operating system
- Architecture (32-bit or 64-bit)
- Launch permissions
- Storage for the root device
- Virtualization Type





Credentials

Used to access an Instance.


Amazon EC2 uses public-key cryptography to encrypt and decrypt login information.

Public-key cryptography uses a public key to encrypt a piece of data and an associated private key to decrypt the data.

These two keys together are called a *key pair*.

AWS stores the public key, and the private key is kept by the customer. The private key is essential to acquiring secure access to an instance for the first time.

When launching a Windows instance, Amazon EC2 generates a random password for the local administrator account and encrypts the password using the public key.



Launching EC2 Instance


Procedure to launch EC2 instance in minutes:

Determine the AWS Region in which you want to launch the Amazon EC2 instance.

Launch an Amazon EC2 instance from a pre-configured Amazon Machine Image (AMI).

Choose an instance type based on CPU, memory, storage, and network requirements.

Configure network, security groups, storage volume, tags, and key pair; or directly launch instance post selecting AMI and Instance type.



LAB 3 : Create an EC2 Linux Instance

Console Access: <https://console.aws.amazon.com/>

Observe different available AMIs and Instance types

Exercise:

Create a Linux Operating System using mandatory configurations:

Instance Type : **T2.Micro**

AMI : **Amazon Linux 2 AMI**

Credentials: **Create/download your own key pair and use same.**

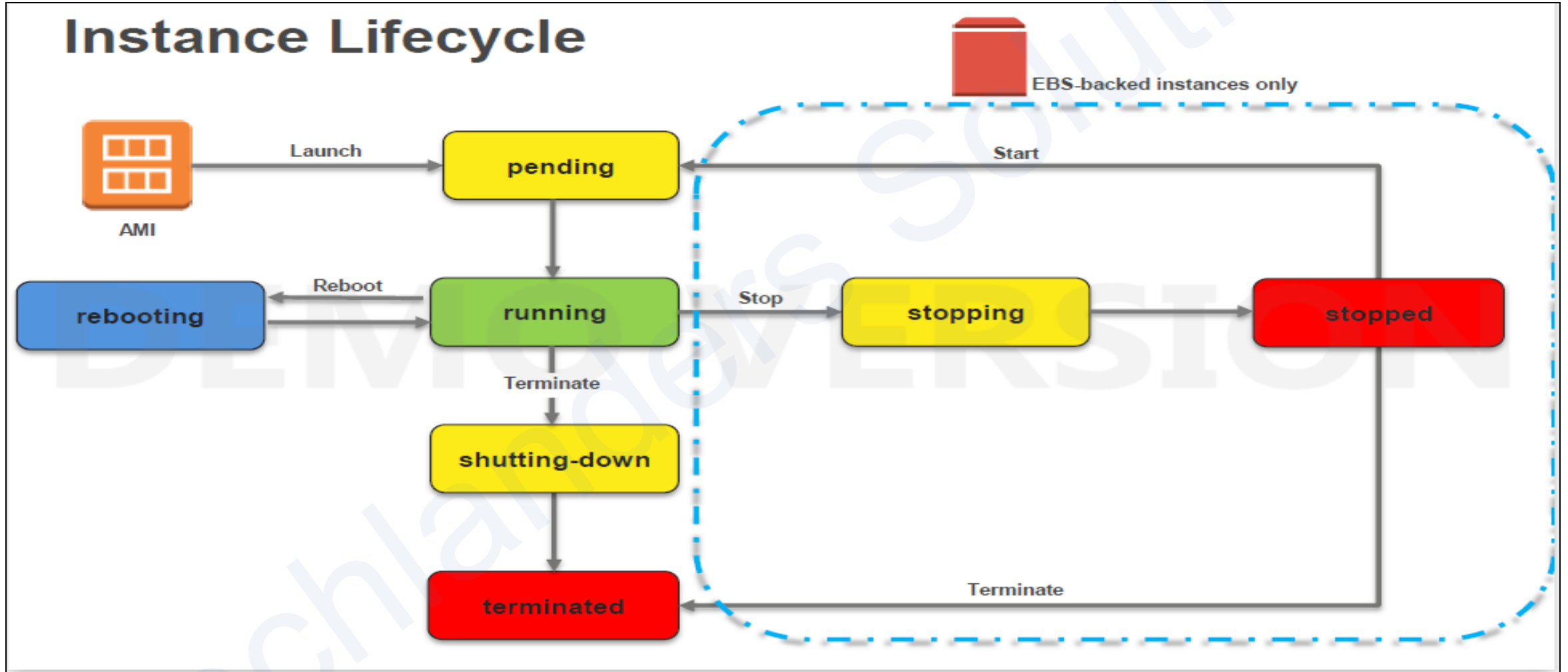
Observe the instance details and navigate through the diff tabs

Reboot, Stop, Start your server

Access your instance using above downloaded key pair using different platforms i.e ,
instance connect , using putty , cli.

Note: (For the first instance take the default parameters wherever inputs required)

Instance Lifecycle




Network part

Private IP: Alike traditional VMs/hosts, each EC2 instance must at least belong to one network (VPC in AWS) and must have at least one Private IP from that network/subnet.

Public IP: A launched instance may also have a public IP address assigned. This IP address is assigned from the addresses reserved by AWS and cannot be specified. This IP address is unique on the Internet, persists only while the instance is running, and cannot be transferred to another instance.

Private/Public Domain Name System (DNS): When you launch an instance, AWS creates one private and one Public DNS name that can be used to access the instance. This DNS name is generated automatically and cannot be specified by the customer. This DNS name persists only while the instance is running and cannot be transferred to another instance.

Elastic IP (EIP): An elastic IP address is an address unique on the Internet that you reserve independently and associate with an Amazon EC2 instance.



EC2 Security – Security Group

Virtual Firewall Protection

AWS allows you to control traffic **in** and **out** of your instances through virtual firewalls called *security groups*.

Security groups allow you to control traffic based on *port, protocol, and source(inbound)/destination(outbound)*.

Security groups are associated with instances when they are launched. **Every instance must have at least one security group.** Though they can have more.

A security group is default deny.



Amazon Server Storage

Amazon EBS (Elastic Block Store)

- Data stored on an Amazon EBS volume can persist independently of the life of the instance.
- Persistent **block-level storage volumes** for use with Amazon EC2 instances.
- 99% used volume type in AWS.

Amazon EC2 Instance Store

- Like Swap volumes attached to your server for faster processing.
- Data stored on a local instance store persists only as long as the instance is alive.
- Storage is ephemeral.
- Older volume type, used exceptionally in some instances i.e DB, High processing systems.

AWS Tags

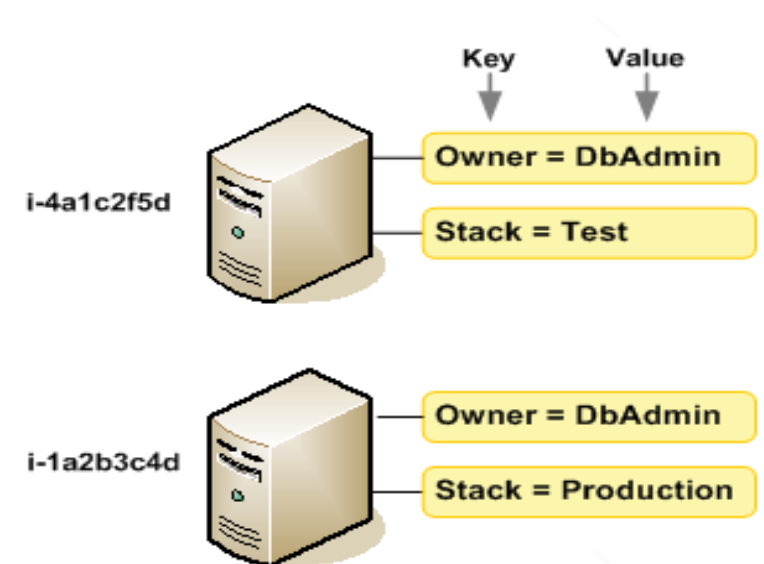
Tags help you manage your instances, images, and other Amazon EC2 resources.

It's your own metadata to each resource in the form of *tags*.

Each tag consists of a *key* and an optional *value*, both of which you define.

It enables you to categorize your AWS resources in different ways, for example, by purpose, owner, or environment.

You can organize your billing information based on resources that have the same tag key values.



AWS Tags

Maximum number of tags per resource—50

Tag keys and values are case-sensitive.

You can't terminate, stop, or delete a resource based solely on its tags; you must specify the resource identifier : ARN's

Not all resources supports tags. Pls check the supported list from AWS website.



EC2 Metadata

AWS Instance Metadata –

Instance metadata is data/information about your instance.

An HTTP call to **`http://169.254.169.254/latest/meta-data/`** will return the top node of the instance metadata tree and will return the information about your instance e.g. Public IP, DNS name, OS type etc..

For Linux systems:

curl <http://169.254.169.254/latest/meta-data/>

Dynamic Instance Identity metadata:

<http://169.254.169.254/latest/dynamic/instance-identity/>

LAB 4 : Create Windows Instance

Exercise:

Create a Windows Operating System using custom configurations:

Instance Type : **T2.Micro**

AMI : **Microsoft Windows Server 2019 Base**

Credentials: **Use existing key from Lab 3 or if not create a new one.**

Assign tags, **Key = Name, Value= Server-Name**

Assign tags, **Key = Owner, Value= Your name**

Add inbound port of **RDP**

Access your instance using Administrator password, which will be retrieved using your downloaded key pair using mstsc or dns

AWS CLI

AWS CLI is a command based utility to manage AWS resources

The primary distribution method for the AWS CLI on Linux, Windows, and macOS is pip, a package manager for Python that provides an easy way to install, upgrade, and remove Python packages and their dependencies

<http://docs.aws.amazon.com/cli/latest/userguide/installing.html>

Requirements

- Python 2 version 2.6.5+ or Python 3 version 3.3+

- Windows, Linux, macOS, or Unix

- Pip package should be present (else install python-pip)

Install AWSCLI: **pip install awscli --upgrade --user**

For Windows, directly download the Windows installer from CLI webpage

LAB 2 : Install AWS CLI

Lets install an AWSCLI

<https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

aws --version

aws help

aws ec2 help / aws s3 help / aws <anysubcommand> help

Configure your default keys and region:

```
root@ip-172-31-28-145:~# aws configure
AWS Access Key ID [None]: #####
AWS Secret Access Key [None]: #####
Default region name [None]: us-west-2
Default output format [None]:
root@ip-172-31-28-145:~#
```

LAB 2 : Install AWS CLI

Watch Credentials and configurations in .aws directory under current directory being created/updated with aws configure:

```
C:\Users\gd>dir .aws
Volume in drive C is System
Volume Serial Number is 5205-3B4A
Directory of C:\Users\CQBJ1454\.aws
2018-02-11  19:13    <DIR>        .
2018-02-11  19:13    <DIR>        ..
2018-02-11  19:29                48 config
2018-05-05  09:03               119 credentials
                2 File(s)        167 bytes
                2 Dir(s)  452,272,128 bytes free
C:\Users\gd>
```


LAB 6 : Creating server using CLI

Check the details for all running instances using CLI

```
aws ec2 describe-instances | grep -i instanceid
```

Creation of an AWS Instance using CLI:

```
aws ec2 run-instances --image-id ami-033b95fb8079dc481 --instance-type t2.micro  
--key-name virginia-key
```

```
aws ec2 describe-instances | grep -i instanceid
```

```
aws ec2 stop-instances --instance-ids i-052af6a166198df10
```

```
aws ec2 terminate-instances --instance-ids i-052af6a166198df10
```



Requirement 1:

Order history app



Requirement 2:

Product recommendations



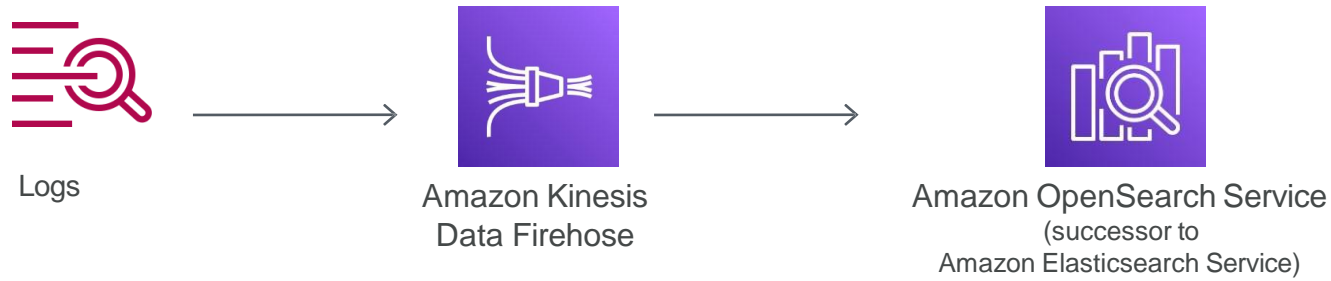
Requirement 3:

Transaction rate alarm

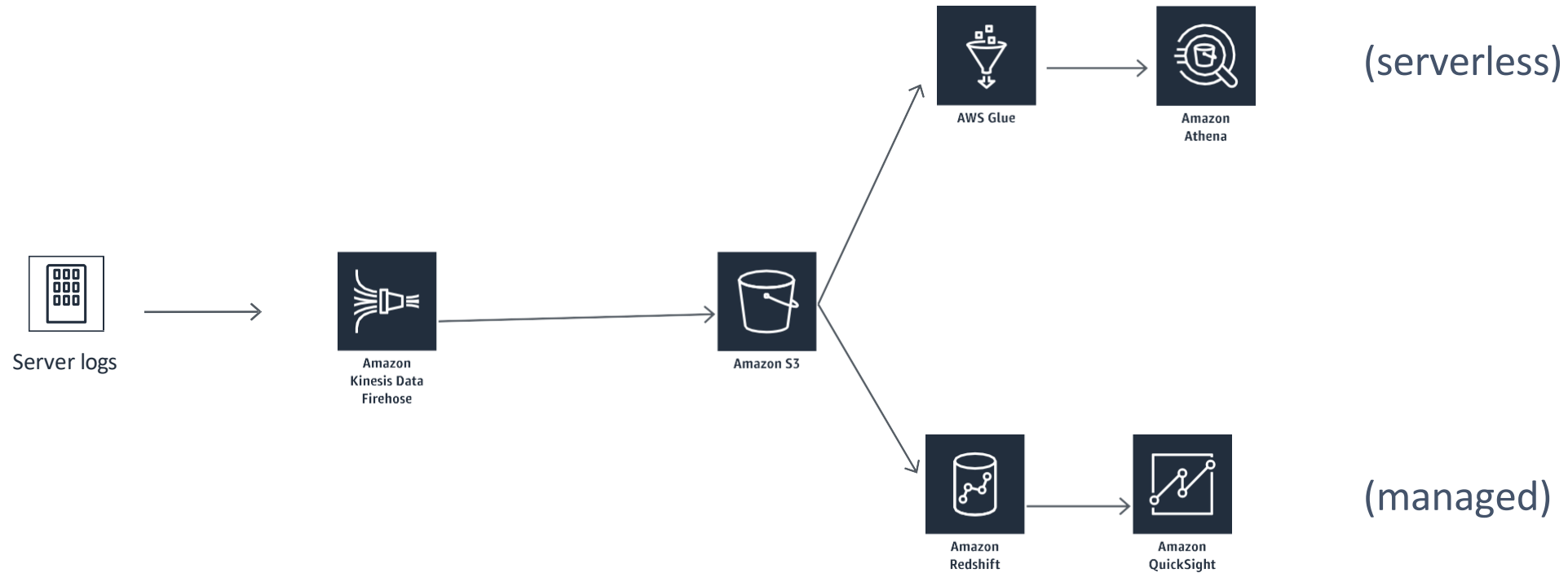


Requirement 4:

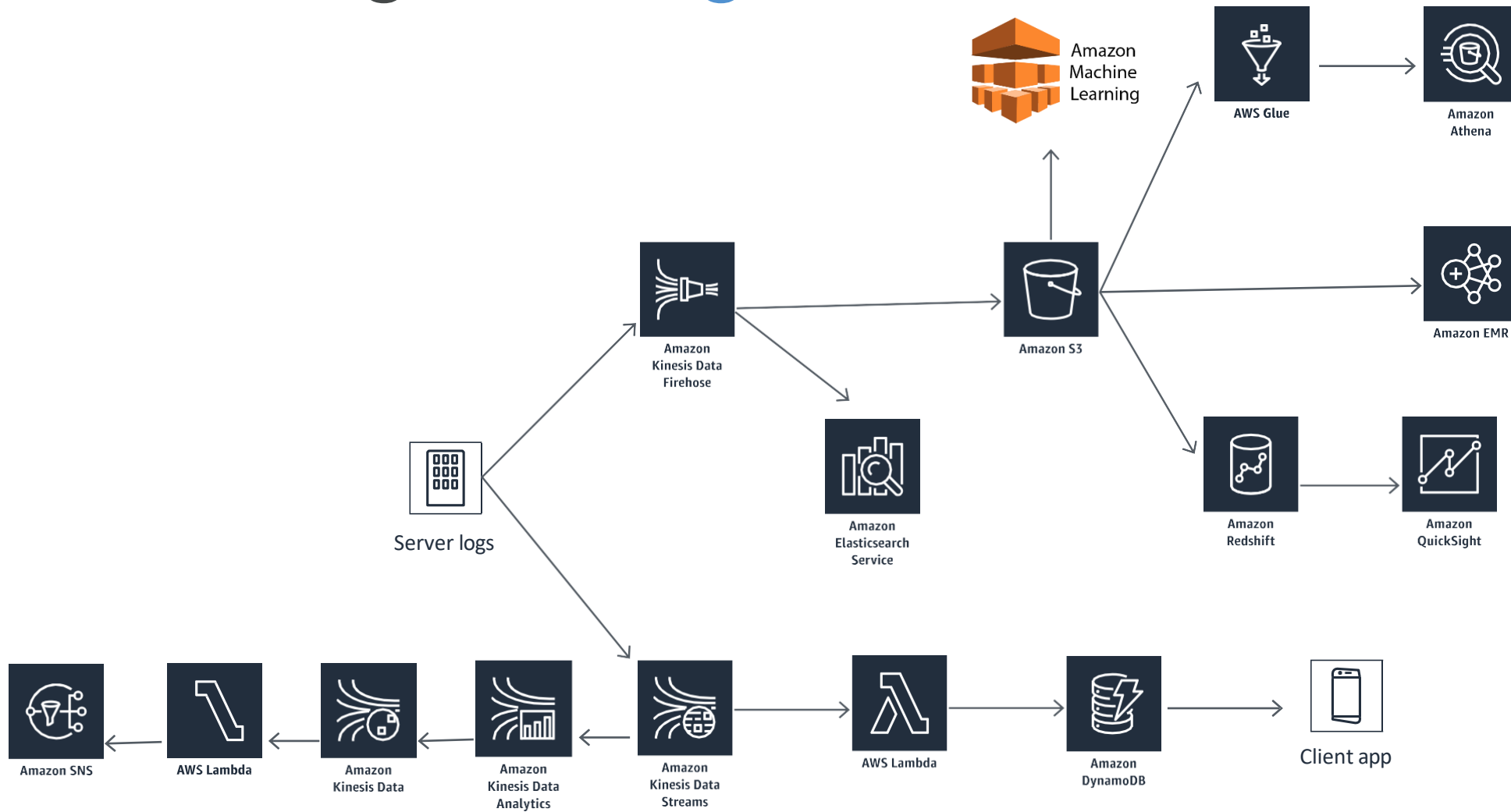
Near-real-time log analysis



Requirement 5: Data warehousing & visualization




Putting it *all* together



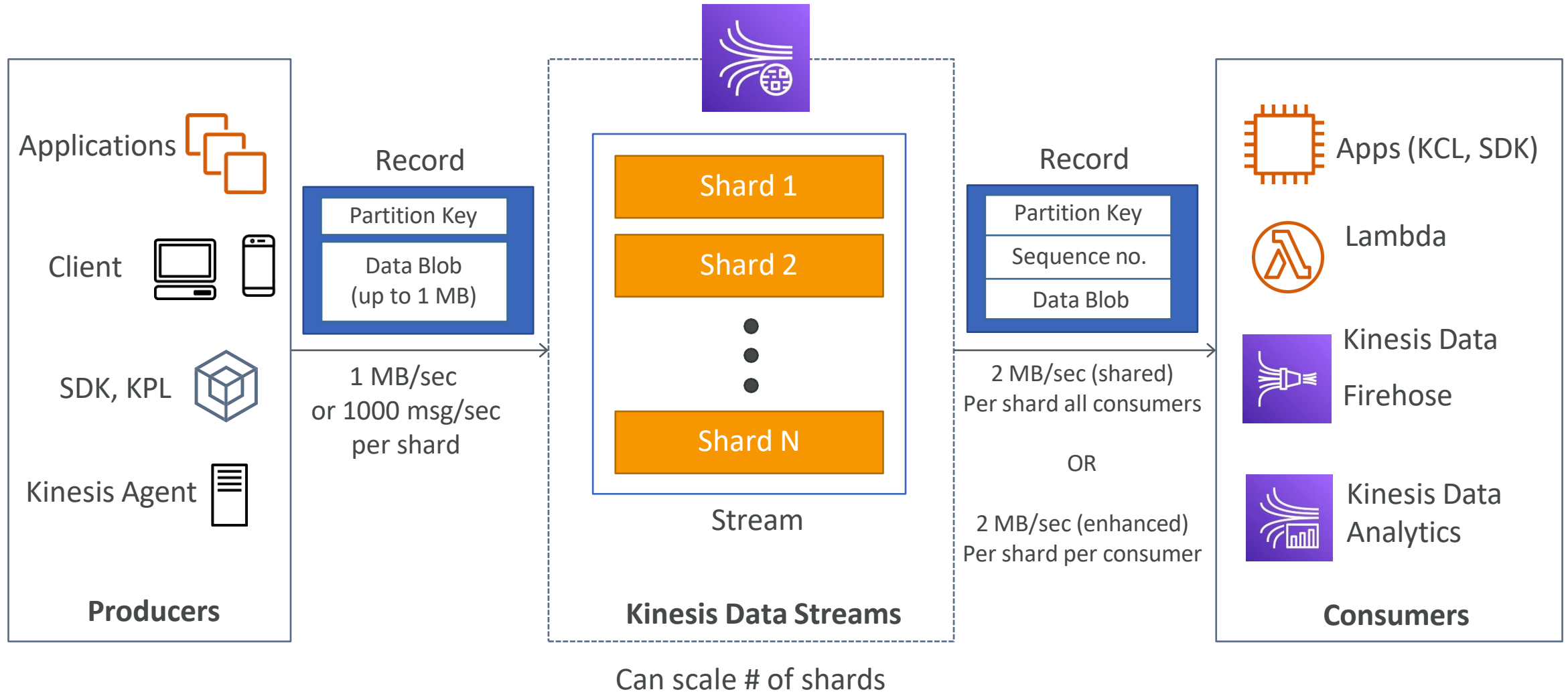
Collection

Moving data into AWS

Collection Introduction

- Real Time - Immediate actions
 - Kinesis Data Streams (KDS)
 - Simple Queue Service (SQS)
 - Internet of Things (IoT)
 - Near-real time - Reactive actions
 - Kinesis Data Firehose (KDF)
 - Database Migration Service (DMS)
- 

Kinesis Data Streams



Kinesis Data Streams



- Retention between 1 day to 365 days
- Ability to reprocess (replay) data
- Once data is inserted in Kinesis, it can't be deleted (immutability)
- Data that shares the same partition goes to the same shard (ordering)
- Producers: AWS SDK, Kinesis Producer Library (KPL), Kinesis Agent
- Consumers:
 - Write your own: Kinesis Client Library (KCL), AWS SDK
 - Managed: AWS Lambda, Kinesis Data Firehose, Kinesis Data Analytics,

Kinesis Data Streams – Capacity Modes

- **Provisioned mode:**

- You choose the number of shards provisioned, scale manually or using API
- Each shard gets 1MB/s in (or 1000 records per second)
- Each shard gets 2MB/s out (classic or enhanced fan-out consumer)
- You pay per shard provisioned per hour

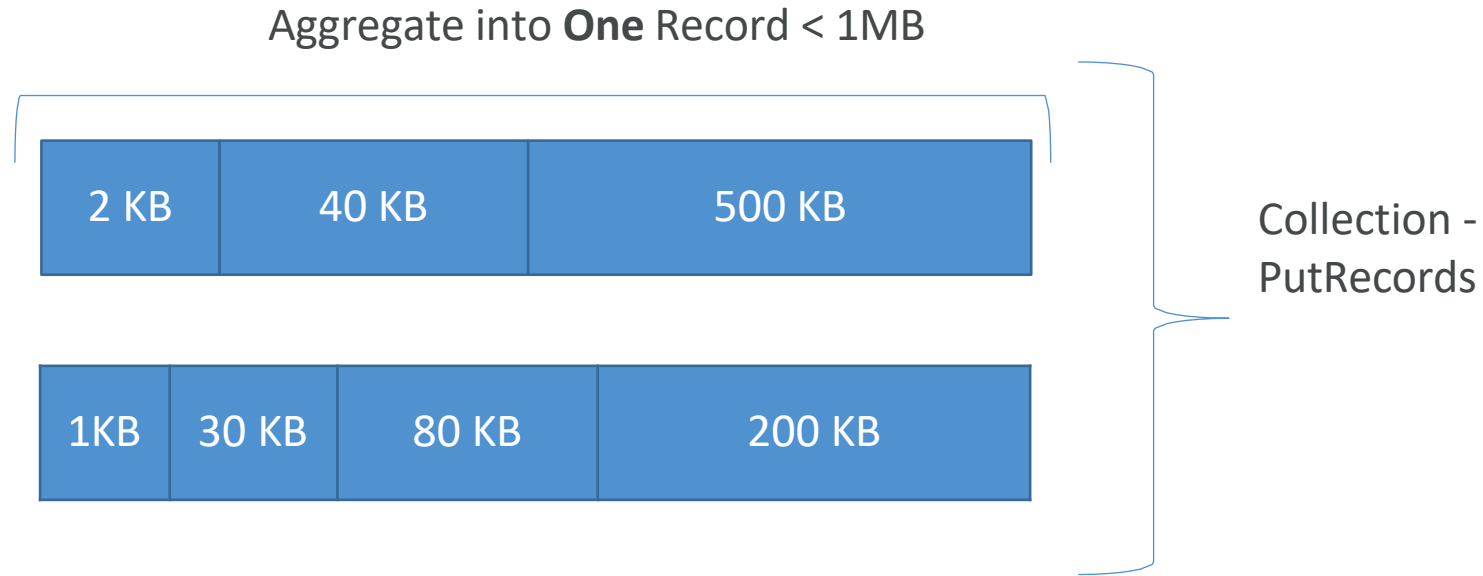
- **On-demand mode:**

- No need to provision or manage the capacity
- Default capacity provisioned (4 MB/s in or 4000 records per second)
- Scales automatically based on observed throughput peak during the last 30 days
- Pay per stream per hour & data in/out per GB

AWS Kinesis API – Exceptions

- ProvisionedThroughputExceeded Exceptions
 - Happens when sending more data (exceeding MB/s or TPS for any shard)
 - Make sure you don't have a hot shard (such as your partition key is bad and too much data goes to that partition)
- Solution:
 - Retries with backoff
 - Increase shards (scaling)
 - Ensure your partition key is a good one

Kinesis Producer Library (KPL) Batching



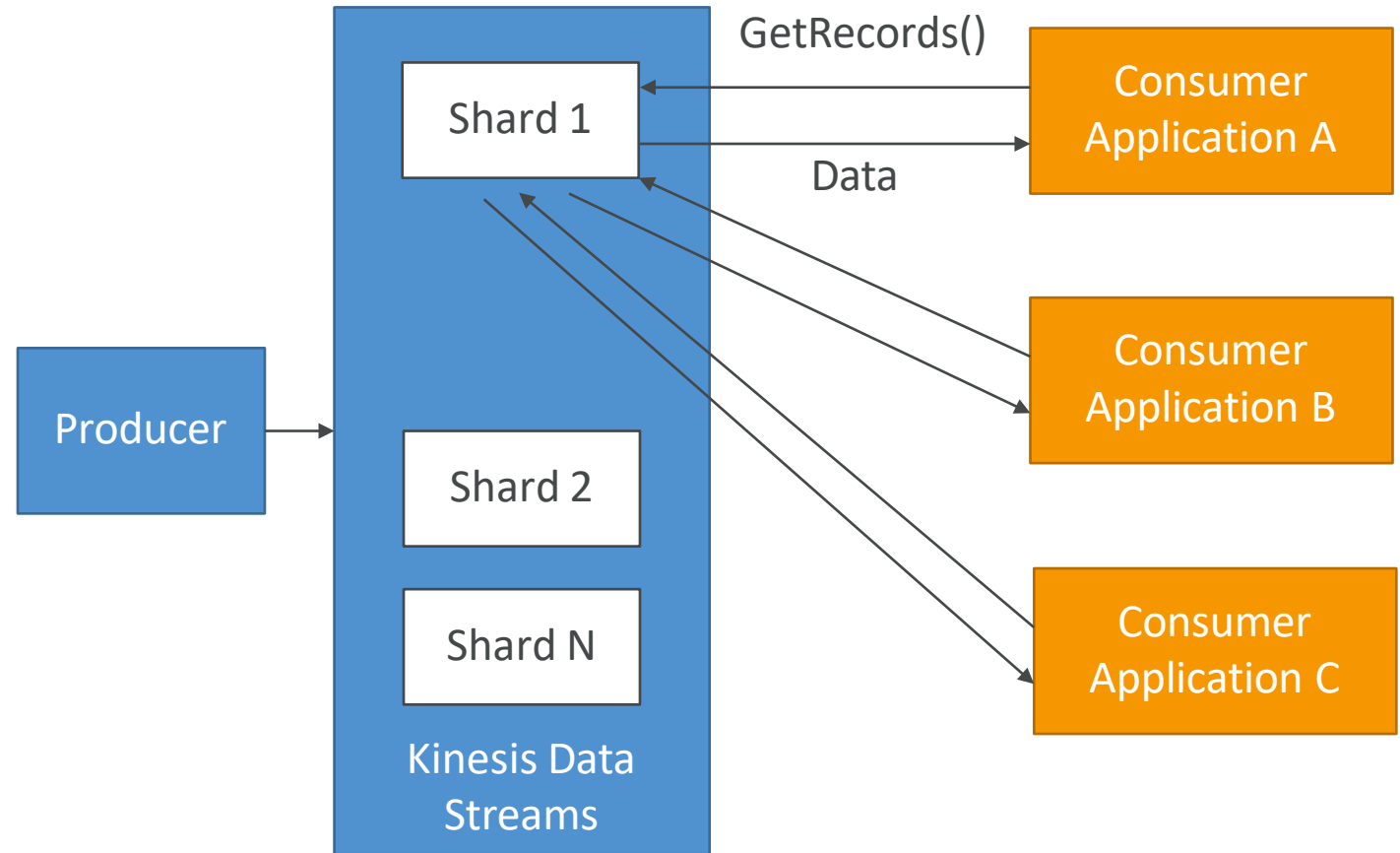
- We can influence the batching efficiency by introducing some delay with `RecordMaxBufferedTime` (default 100ms)

Kinesis Agent

- Monitor Log files and sends them to Kinesis Data Streams
- Java-based agent, built on top of KPL
- Install in Linux-based server environments
- Features:
 - Write from multiple directories and write to multiple streams
 - Routing feature based on directory / log file
 - Pre-process data before sending to streams (single line, csv to json, log to json...)
 - The agent handles file rotation, checkpointing, and retry upon failures
 - Emits metrics to CloudWatch for monitoring

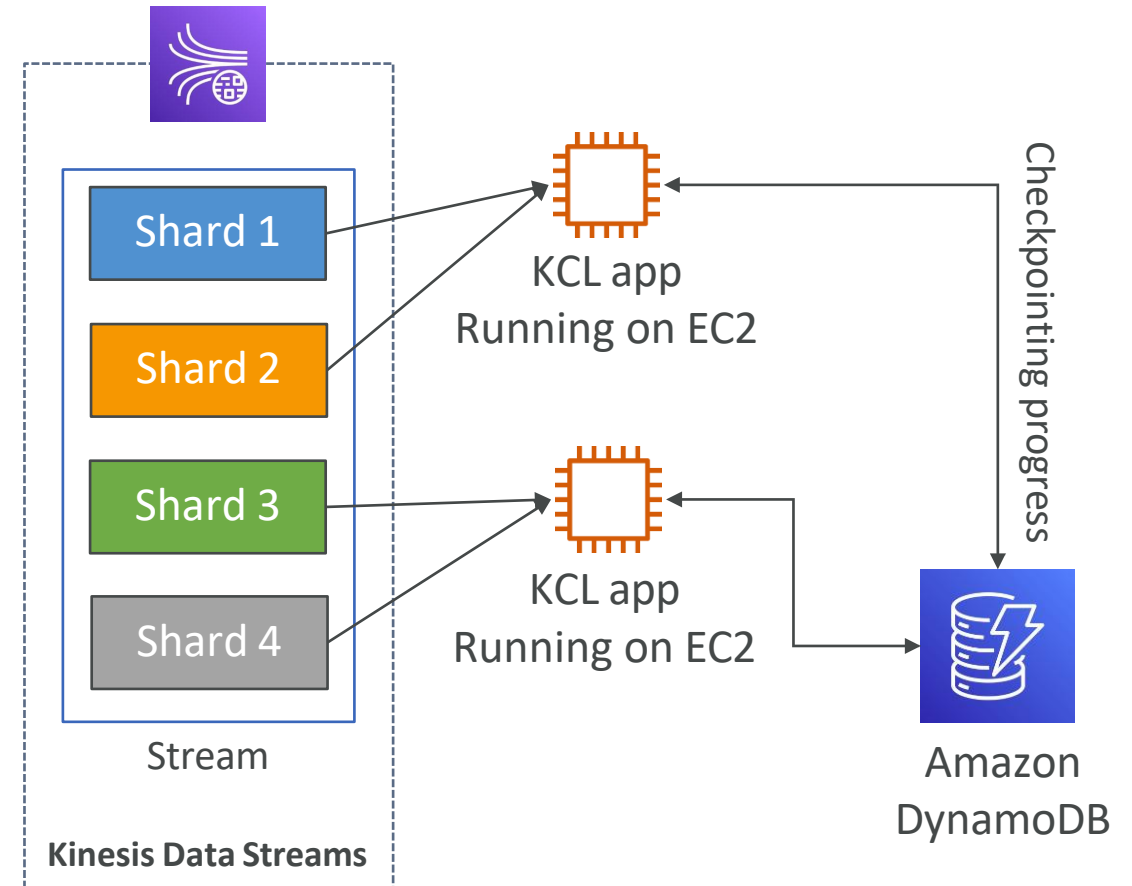
Kinesis Consumer SDK - GetRecords

- **Classic Kinesis** - Records are polled by consumers from a shard
- **Each shard has 2 MB total aggregate throughput**
- GetRecords returns up to 10MB of data (then throttle for 5 seconds) or up to 10000 records
- Maximum of 5 GetRecords API calls per shard per second = 200ms latency
- If 5 consumers application consume from the same shard, means every consumer can poll once a second and receive less than 400 KB/s



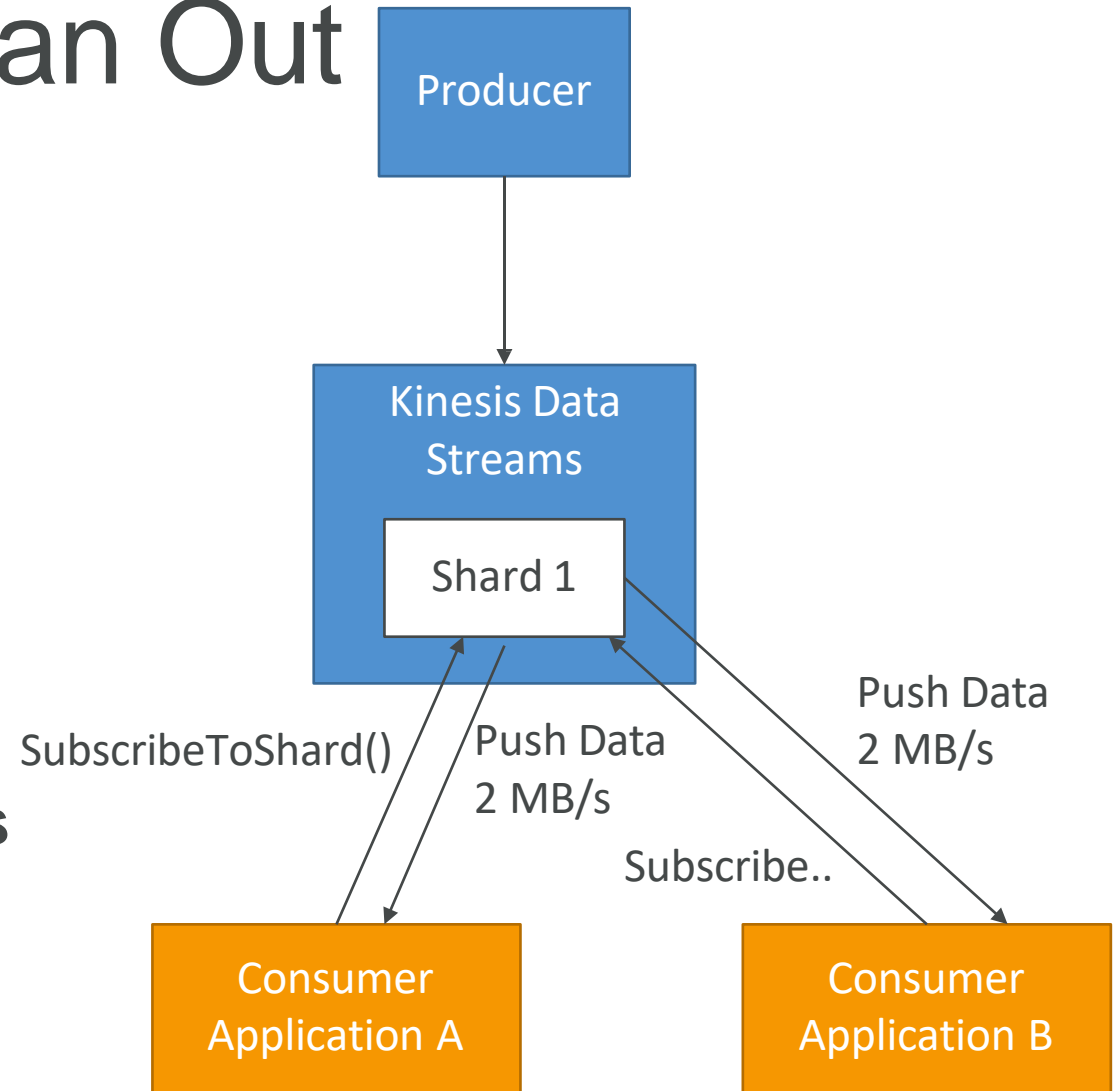
Kinesis Client Library (KCL)

- Java-first library but exists for other languages too (Golang, Python, Ruby, Node, .NET ...)
- Read records from Kinesis produced with the KPL (de-aggregation)
- Share multiple shards with multiple consumers in one “group”, **shard discovery**
- **Checkpointing** feature to resume progress
- Leverages DynamoDB for coordination and checkpointing (one row per shard)
 - Make sure you provision enough WCU / RCU
 - Or use On-Demand for DynamoDB
 - Otherwise DynamoDB may slow down KCL
- Record processors will process the data
- **ExpiredIteratorException => increase WCU**



Kinesis Enhanced Fan Out

- New **game-changing** feature from August 2018.
- Works with KCL 2.0 and AWS Lambda (Nov 2018)
- Each Consumer get 2 MB/s of provisioned throughput per shard
- That means 20 consumers will get 40MB/s per shard aggregated
- No more 2 MB/s limit!
- Enhanced Fan Out: Kinesis **pushes** data to consumers over HTTP/2
- Reduce latency (~70 ms)

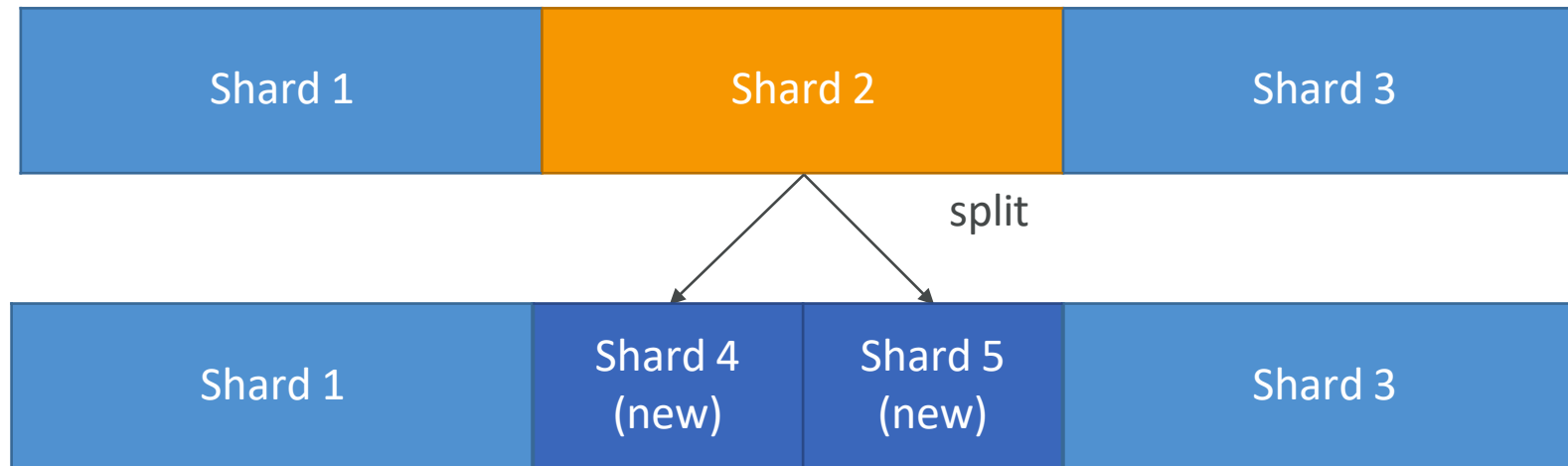


Enhanced Fan-Out vs Standard Consumers

- Standard consumers:
 - Low number of consuming applications (1,2,3...)
 - Can tolerate ~200 ms latency
 - Minimize cost
- Enhanced Fan Out Consumers:
 - Multiple Consumer applications for the same Stream
 - Low Latency requirements ~70ms
 - Higher costs (see Kinesis pricing page)
 - Default limit of 20 consumers using enhanced fan-out per data stream

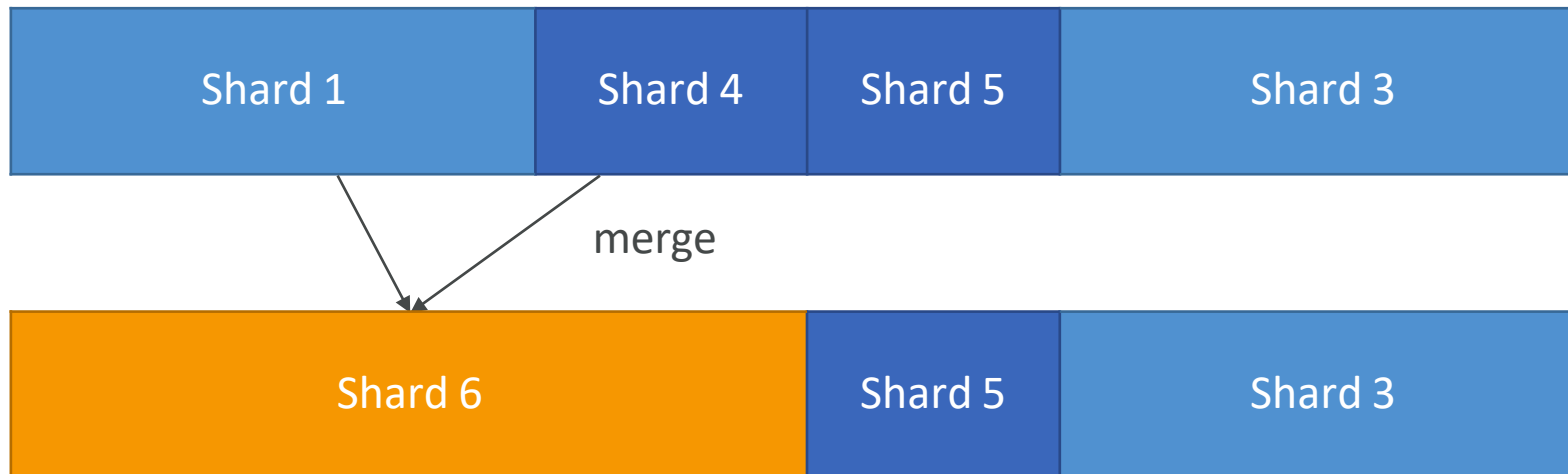
Kinesis Operations – Adding Shards

- Also called “Shard Splitting”
- Can be used to increase the Stream capacity (1 MB/s data in per shard)
- Can be used to divide a “hot shard”
- The old shard is closed and will be deleted once the data is expired



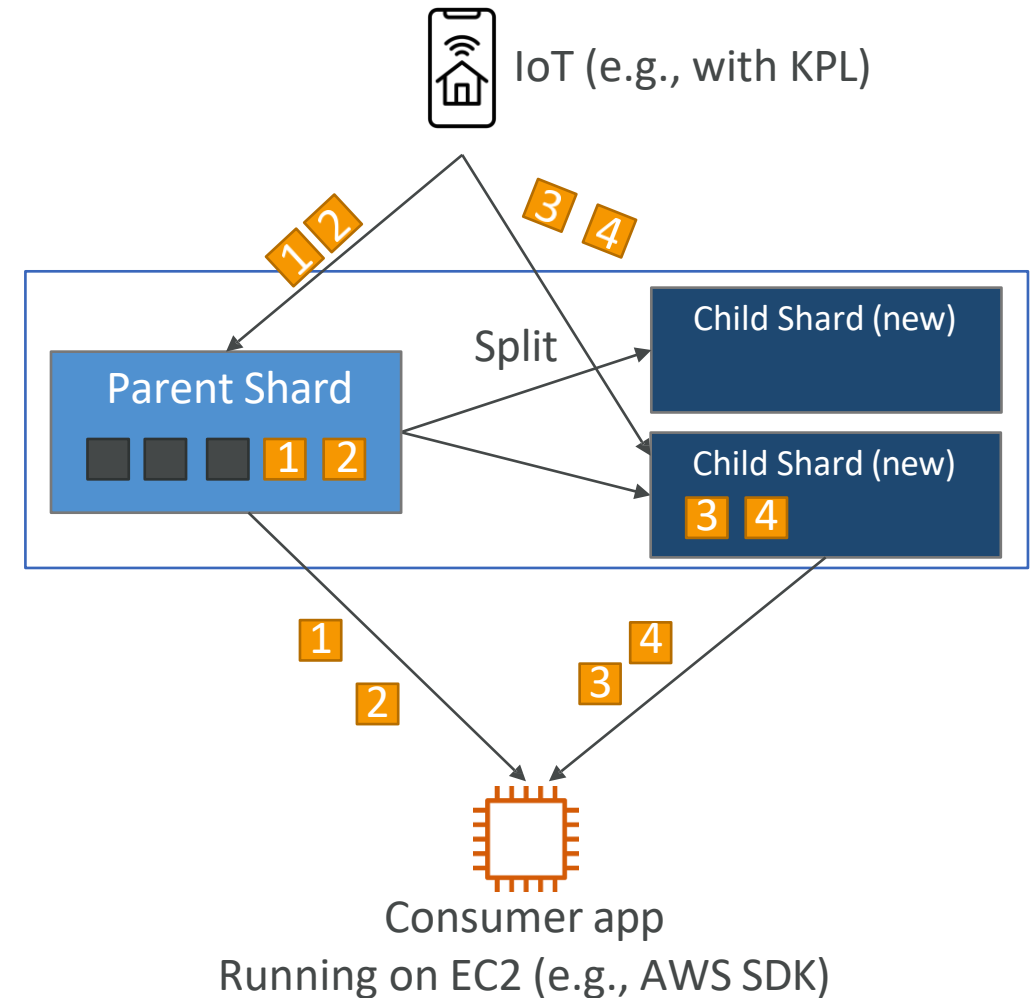
Kinesis Operations – Merging Shards

- Decrease the Stream capacity and save costs
- Can be used to group two shards with low traffic
- Old shards are closed and deleted based on data expiration



Out-of-order records after resharding

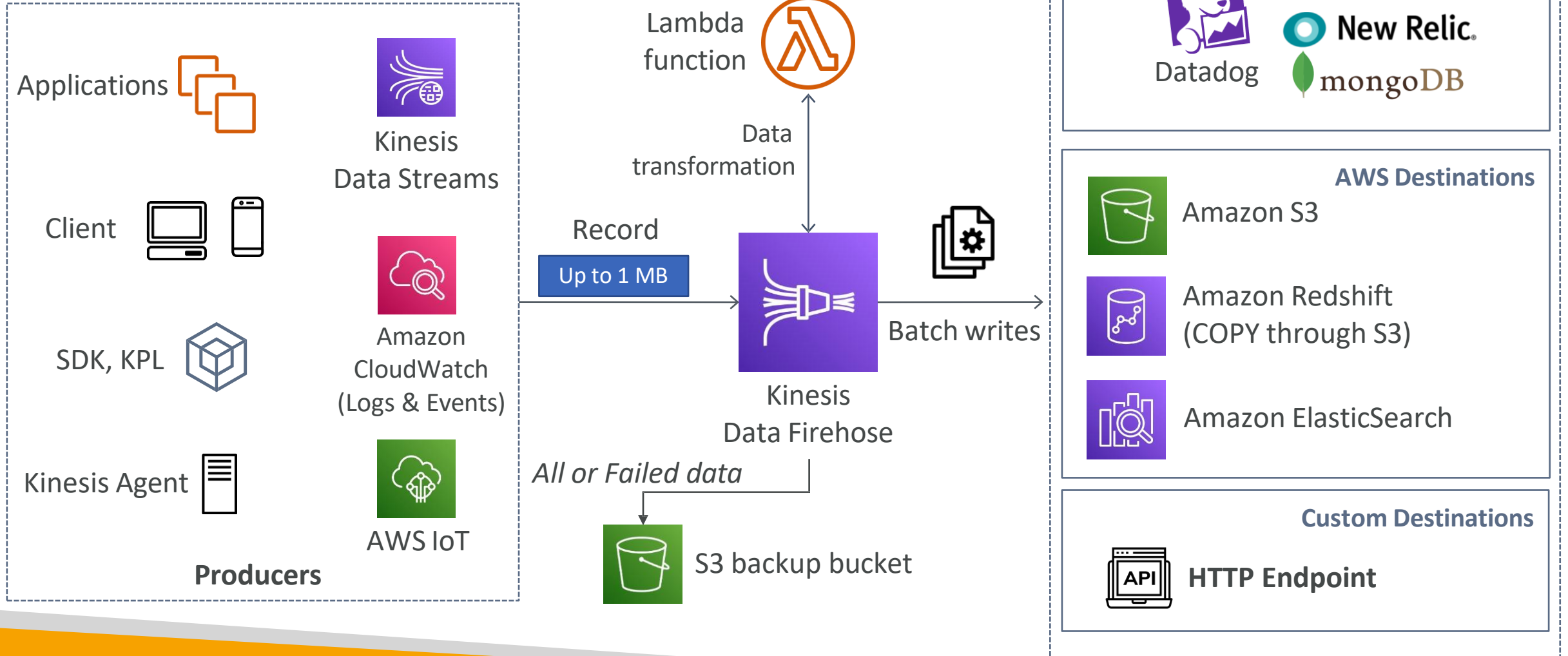
- After a reshard, you can read from child shards
- However, data you haven't read yet could still be in the parent
- If you start reading the child before completing reading the parent, **you could read data for a particular hash key out of order**
- After a reshard, read entirely from the parent until you don't have new records
- Note: The Kinesis Client Library (KCL) has this logic already built-in, even after resharding operations



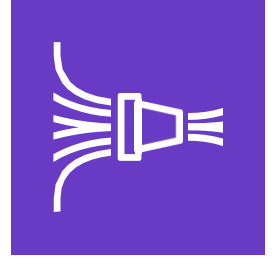
LAB

Kinesis Data Stream usage

Kinesis Data Firehose

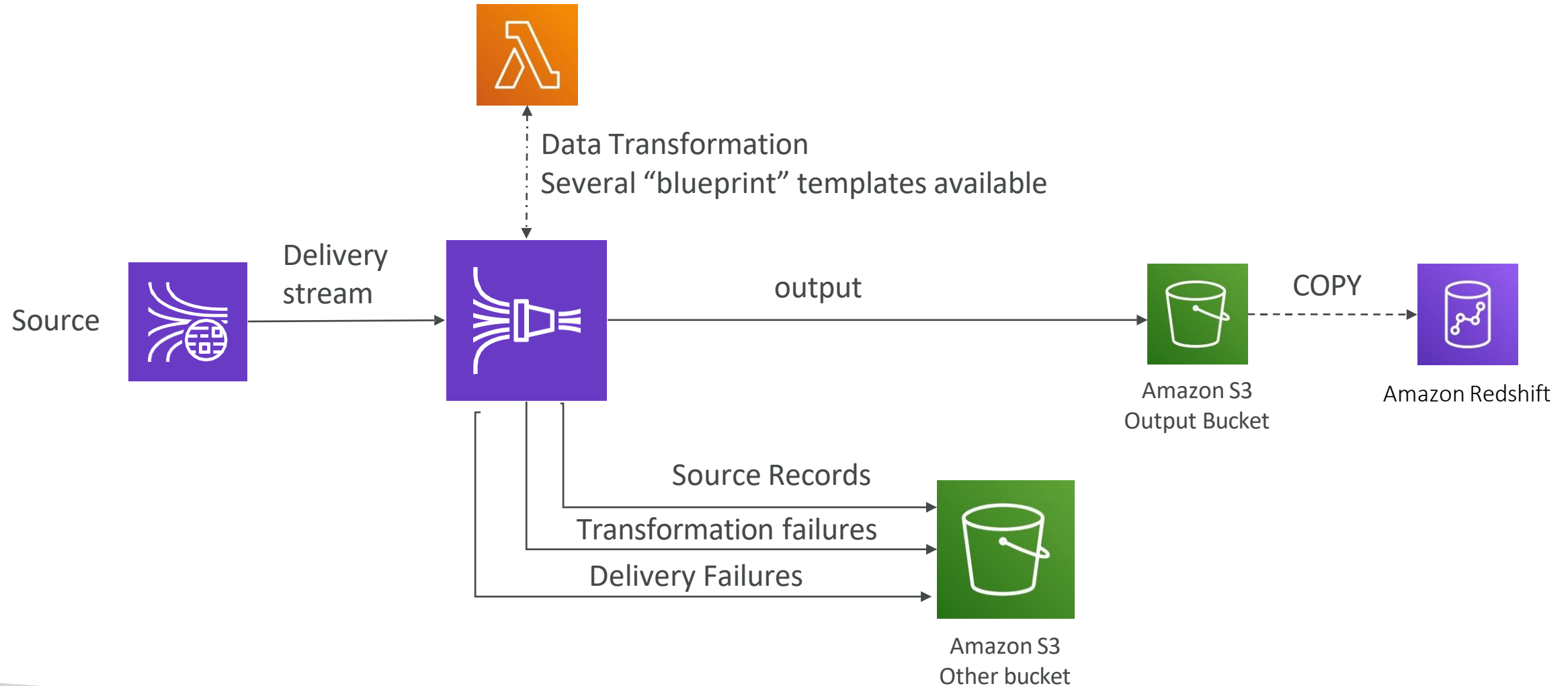


AWS Kinesis Data Firehose



- Fully Managed Service, no administration
- **Near Real Time** (60 seconds latency minimum for non full batches)
- Load data into Redshift / Amazon S3 / ElasticSearch / Splunk
- Automatic scaling
- Supports many data formats
- Data Conversions from JSON to Parquet / ORC (only for S3)
- Data Transformation through AWS Lambda (ex: CSV => JSON)
- Supports **compression** when target is Amazon S3 (GZIP, ZIP, and SNAPPY)
- Only GZIP is the data is further loaded into Redshift
- Pay for the amount of data going through Firehose
- Spark / KCL do not read from KDF

Kinesis Data Firehose Delivery Diagram



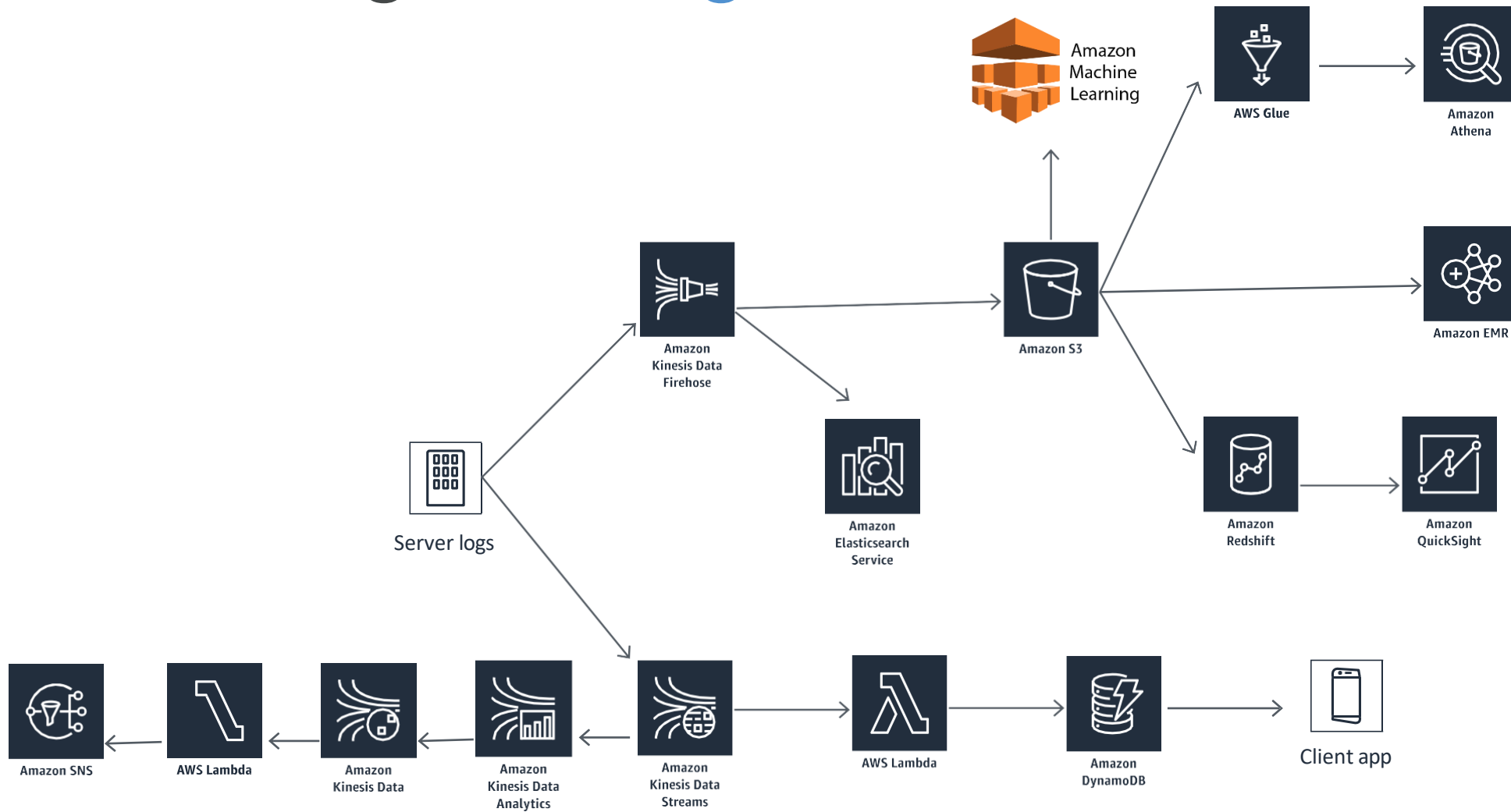
Firehose Buffer Sizing

- Firehose accumulates records in a buffer
- The buffer is flushed based on time and size rules
- Buffer Size (ex: 32MB): if that buffer size is reached, it's flushed
- Buffer Time (ex: 2 minutes): if that time is reached, it's flushed
- Firehose can automatically increase the buffer size to increase throughput
- High throughput => Buffer Size will be hit
- Low throughput => Buffer Time will be hit

Kinesis Data Streams vs Firehose

- Streams
 - Going to write custom code (producer / consumer)
 - Real time (~200 ms latency for classic, ~70 ms latency for enhanced fan-out)
 - Must manage scaling (shard splitting / merging)
 - Data Storage for 1 to 365 days, replay capability, multi consumers
 - Use with Lambda to insert data in real-time to ElasticSearch (for example)
- Firehose
 - Fully managed, send to S3, Splunk, Redshift, ElasticSearch
 - Serverless data transformations with Lambda
 - **Near** real time (lowest buffer time is 1 minute)
 - Automated Scaling
 - No data storage

Putting it *all* together



LAB

ServerLogs – KinesisDataFirehose – S3



Storage

Amazon S3 Section

Section introduction



- Amazon S3 is one of the main building blocks of AWS
- It's advertised as "infinitely scaling" storage
- Many websites use Amazon S3 as a backbone
- Many AWS services use Amazon S3 as an integration as well
- We'll have a step-by-step approach to S3

Amazon S3 Use cases

- Backup and storage
- Disaster Recovery
- Archive
- Hybrid Cloud storage
- Application hosting
- Media hosting
- Data lakes & big data analytics
- Software delivery
- Static website



Nasdaq stores 7 years of data into S3 Glacier



Sysco runs analytics on its data and gain business insights

Amazon S3 - Buckets

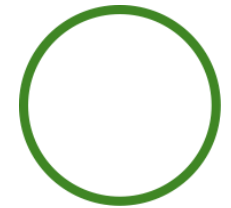
- Amazon S3 allows people to store objects (files) in “buckets” (directories)
- Buckets must have a **globally unique name (across all regions all accounts)**
- Buckets are defined at the region level
- S3 looks like a global service but buckets are created in a region
- Naming convention
 - No uppercase, No underscore
 - 3-63 characters long
 - Not an IP
 - Must start with lowercase letter or number
 - Must NOT start with the prefix **xn--**
 - Must NOT end with the suffix **-s3alias**



S3 Bucket

Amazon S3 - Objects

- Objects (files) have a Key
- The **key** is the **FULL** path:
 - s3://my-bucket/**my_file.txt**
 - s3://my-bucket/**my_folder1/another_folder/my_file.txt**
- The key is composed of **prefix** + **object name**
 - s3://my-bucket/**my_folder1/another_folder/my_file.txt**
- There's no concept of “directories” within buckets (although the UI will trick you to think otherwise)
- Just keys with very long names that contain slashes (“/”)



Object



S3 Bucket
with Objects

Amazon S3 – Objects (cont.)



- Object values are the content of the body:
 - Max. Object Size is 5TB (5000GB)
 - If uploading more than 5GB, must use “multi-part upload”
- Metadata (list of text key / value pairs – system or user metadata)
- Tags (Unicode key / value pair – up to 10) – useful for security / lifecycle
- Version ID (if versioning is enabled)

S3 Bucket Policies

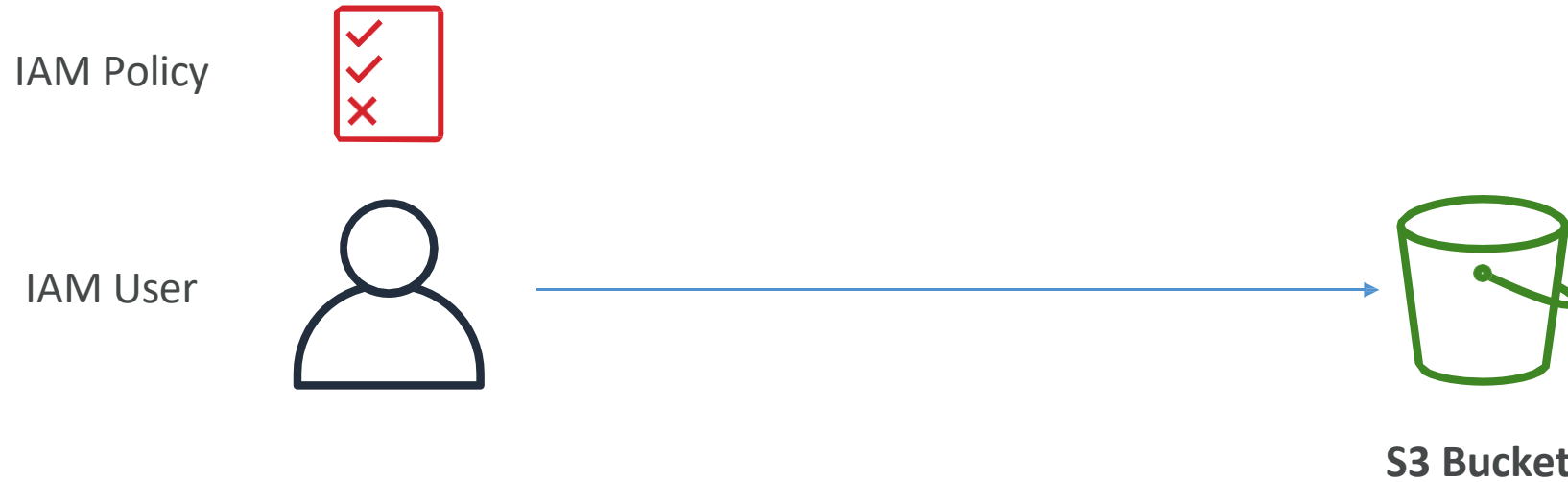
- JSON based policies
 - Resources: buckets and objects
 - Effect: Allow / Deny
 - Actions: Set of API to Allow or Deny
 - Principal: The account or user to apply the policy to
- Use S3 bucket for policy to:
 - Grant public access to the bucket
 - Force objects to be encrypted at upload
 - Grant access to another account (Cross Account)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicRead",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/*"
      ]
    }
  ]
}
```

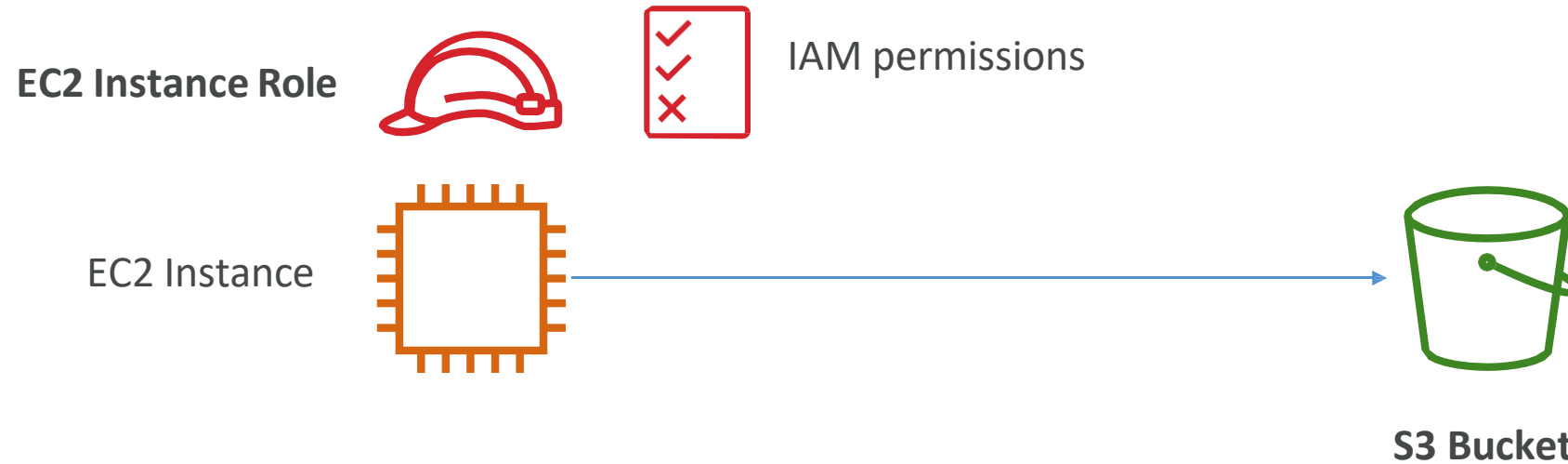
Example: Public Access - Use Bucket Policy



Example: User Access to S3 – IAM permissions



Example: EC2 instance access - Use IAM Roles



Bucket settings for Block Public Access

Block *all* public access

On

Block public access to buckets and objects granted through *new* access control lists (ACLs)

On

Block public access to buckets and objects granted through *any* access control lists (ACLs)

On

Block public access to buckets and objects granted through *new* public bucket or access point policies

On

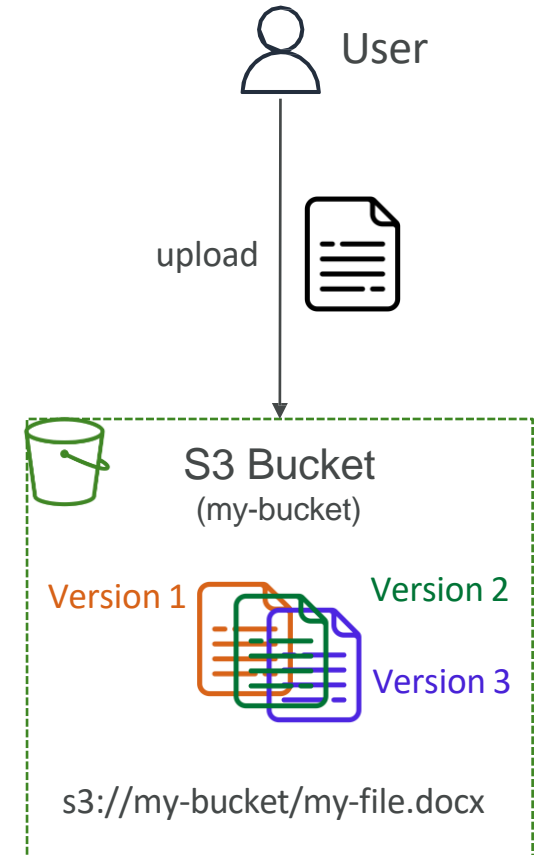
Block public and cross-account access to buckets and objects through *any* public bucket or access point policies

On

- **These settings were created to prevent company data leaks**
- If you know your bucket should never be public, leave these on
- Can be set at the account level

Amazon S3 - Versioning

- You can version your files in Amazon S3
- It is enabled at the **bucket level**
- Same key overwrite will change the “version”: 1, 2, 3....
- It is best practice to version your buckets
 - Protect against unintended deletes (ability to restore a version)
 - Easy roll back to previous version
- Notes:
 - Any file that is not versioned prior to enabling versioning will have version “null”
 - Suspending versioning does not delete the previous versions



S3 Storage Classes

- Amazon S3 Standard - General Purpose
 - Amazon S3 Standard-Infrequent Access (IA)
 - Amazon S3 One Zone-Infrequent Access
 - Amazon S3 Glacier Instant Retrieval
 - Amazon S3 Glacier Flexible Retrieval
 - Amazon S3 Glacier Deep Archive
 - Amazon S3 Intelligent Tiering
-
- Can move between classes manually or using S3 Lifecycle configurations

S3 Durability and Availability

- Durability:
 - High durability (99.999999999%, 11 9's) of objects across multiple AZ
 - If you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years
 - Same for all storage classes
- Availability:
 - Measures how readily available a service is
 - Varies depending on storage class
 - Example: S3 standard has 99.99% availability = not available 53 minutes a year

S3 Standard – General Purpose



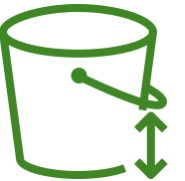
- 99.99% Availability
 - Used for frequently accessed data
 - Low latency and high throughput
 - Sustain 2 concurrent facility failures
-
- Use Cases: Big Data analytics, mobile & gaming applications, content distribution...

S3 Storage Classes – Infrequent Access

- For data that is less frequently accessed, but requires rapid access when needed
- Lower cost than S3 Standard

- **Amazon S3 Standard-Infrequent Access (S3 Standard-IA)**

- 99.9% Availability
- Use cases: Disaster Recovery, backups



- **Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA)**

- High durability (99.9999999999%) in a single AZ; data lost when AZ is destroyed
- 99.5% Availability
- Use Cases: Storing secondary backup copies of on-premises data, or data you can recreate



Amazon S3 Glacier Storage Classes

- Low-cost object storage meant for archiving / backup
- Pricing: price for storage + object retrieval cost
- **Amazon S3 Glacier Instant Retrieval**
 - Millisecond retrieval, great for data accessed once a quarter
 - Minimum storage duration of 90 days
- **Amazon S3 Glacier Flexible Retrieval** (formerly Amazon S3 Glacier):
 - Expedited (1 to 5 minutes), Standard (3 to 5 hours), Bulk (5 to 12 hours) – free
 - Minimum storage duration of 90 days
- **Amazon S3 Glacier Deep Archive – for long term storage:**
 - Standard (12 hours), Bulk (48 hours)
 - Minimum storage duration of 180 days



S3 Intelligent-Tiering



- Small monthly monitoring and auto-tiering fee
- Moves objects automatically between Access Tiers based on usage
- There are no retrieval charges in S3 Intelligent-Tiering
- *Frequent Access tier (automatic)*: default tier
- *Infrequent Access tier (automatic)*: objects not accessed for 30 days
- *Archive Instant Access tier (automatic)*: objects not accessed for 90 days
- *Archive Access tier (optional)*: configurable from 90 days to 700+ days
- *Deep Archive Access tier (optional)*: config. from 180 days to 700+ days

S3 Storage Classes Comparison

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
Availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99%	99.9%	99.9%
Availability Zones	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

<https://aws.amazon.com/s3/storage-classes/>

S3 Storage Classes – Price Comparison

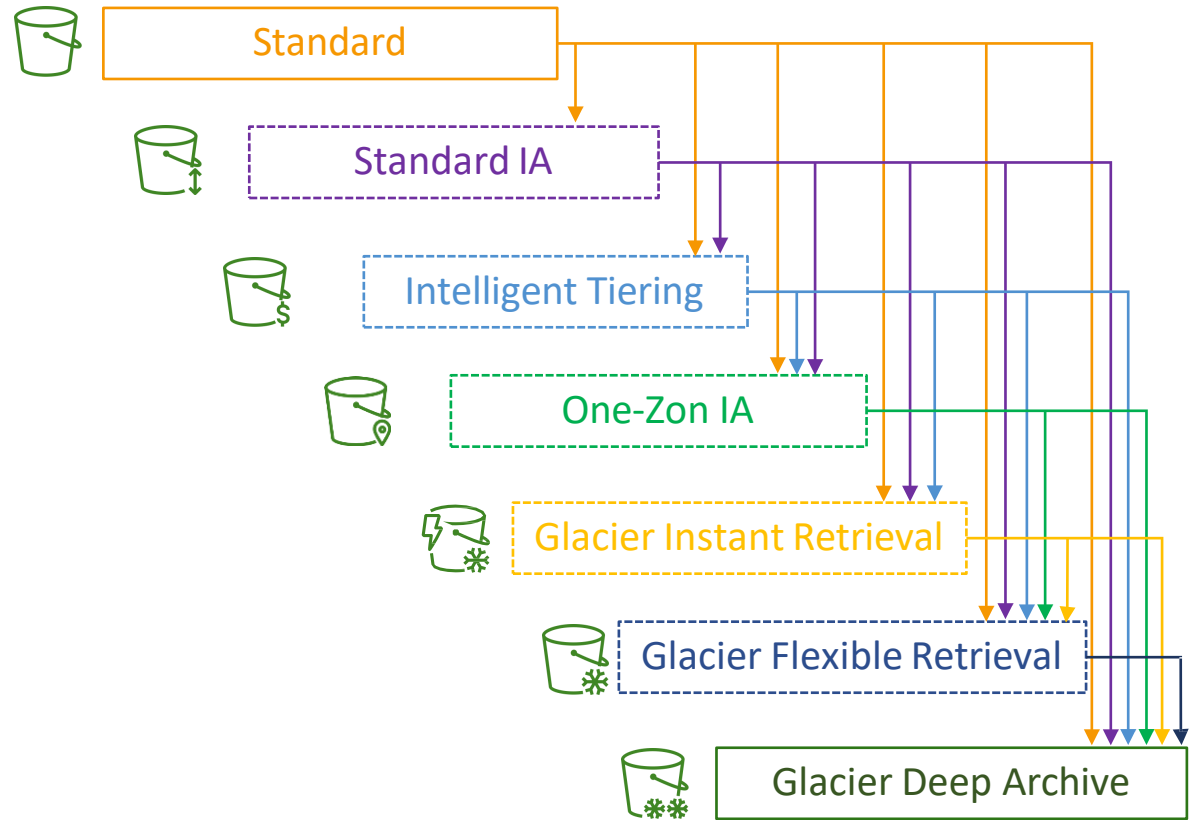
Example: us-east-1

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Storage Cost (per GB per month)	\$0.023	\$0.0025 - \$0.023	\$0.0125	\$0.01	\$0.004	\$0.0036	\$0.00099
Retrieval Cost (per 1000 request)	GET: \$0.0004 POST: \$0.005	GET: \$0.0004 POST: \$0.005	GET: \$0.001 POST: \$0.01	GET: \$0.001 POST: \$0.01	GET: \$0.01 POST: \$0.02	GET: \$0.0004 POST: \$0.03 Expedited: \$10 Standard: \$0.05 Bulk: free	GET: \$0.0004 POST: \$0.05 Standard: \$0.10 Bulk: \$0.025
Retrieval Time	Instantaneous					Expedited (1 – 5 mins) Standard (3 – 5 hours) Bulk (5 – 12 hours)	Standard (12 hours) Bulk (48 hours)
Monitoring Cost (per 1000 objects)		\$0.0025					

<https://aws.amazon.com/s3/pricing/>

Amazon S3 – Moving between Storage Classes

- You can transition objects between storage classes
- For infrequently accessed object, move them to **Standard IA**
- For archive objects that you don't need fast access to, move them to **Glacier or Glacier Deep Archive**
- Moving objects can be automated using a **Lifecycle Rules**



Amazon S3 – Lifecycle Rules



- **Transition Actions** – configure objects to transition to another storage class
 - Move objects to Standard IA class 60 days after creation
 - Move to Glacier for archiving after 6 months
- **Expiration actions** – configure objects to expire (delete) after some time
 - Access log files can be set to delete after a 365 days
 - **Can be used to delete old versions of files (if versioning is enabled)**
 - Can be used to delete incomplete Multi-Part uploads
- Rules can be created for a certain prefix (example: *s3://mybucket/mp3/**)
- Rules can be created for certain objects Tags (example: *Department: Finance*)

Amazon S3 – Lifecycle Rules (Scenario 1)

- Your application on EC2 creates images thumbnails after profile photos are uploaded to Amazon S3. These thumbnails can be easily recreated, and only need to be kept for 60 days. The source images should be able to be immediately retrieved for these 60 days, and afterwards, the user can wait up to 6 hours. How would you design this?
- S3 source images can be on **Standard**, with a lifecycle configuration to transition them to **Glacier** after 60 days
- S3 thumbnails can be on **One-Zone IA**, with a lifecycle configuration to expire them (delete them) after 60 days

Amazon S3 – Lifecycle Rules (Scenario 2)

- A rule in your company states that you should be able to recover your deleted S3 objects immediately for 30 days, although this may happen rarely. After this time, and for up to 365 days, deleted objects should be recoverable within 48 hours.
- **Enable S3 Versioning** in order to have object versions, so that “deleted objects” are in fact hidden by a “delete marker” and can be recovered
- Transition the “noncurrent versions” of the object to **Standard IA**
- Transition afterwards the “noncurrent versions” to **Glacier Deep Archive**

AWS EBS Service


Storage Disk Addition

Storage Addition, Filesystem Creation are very common tasks in day to day activity for any application.

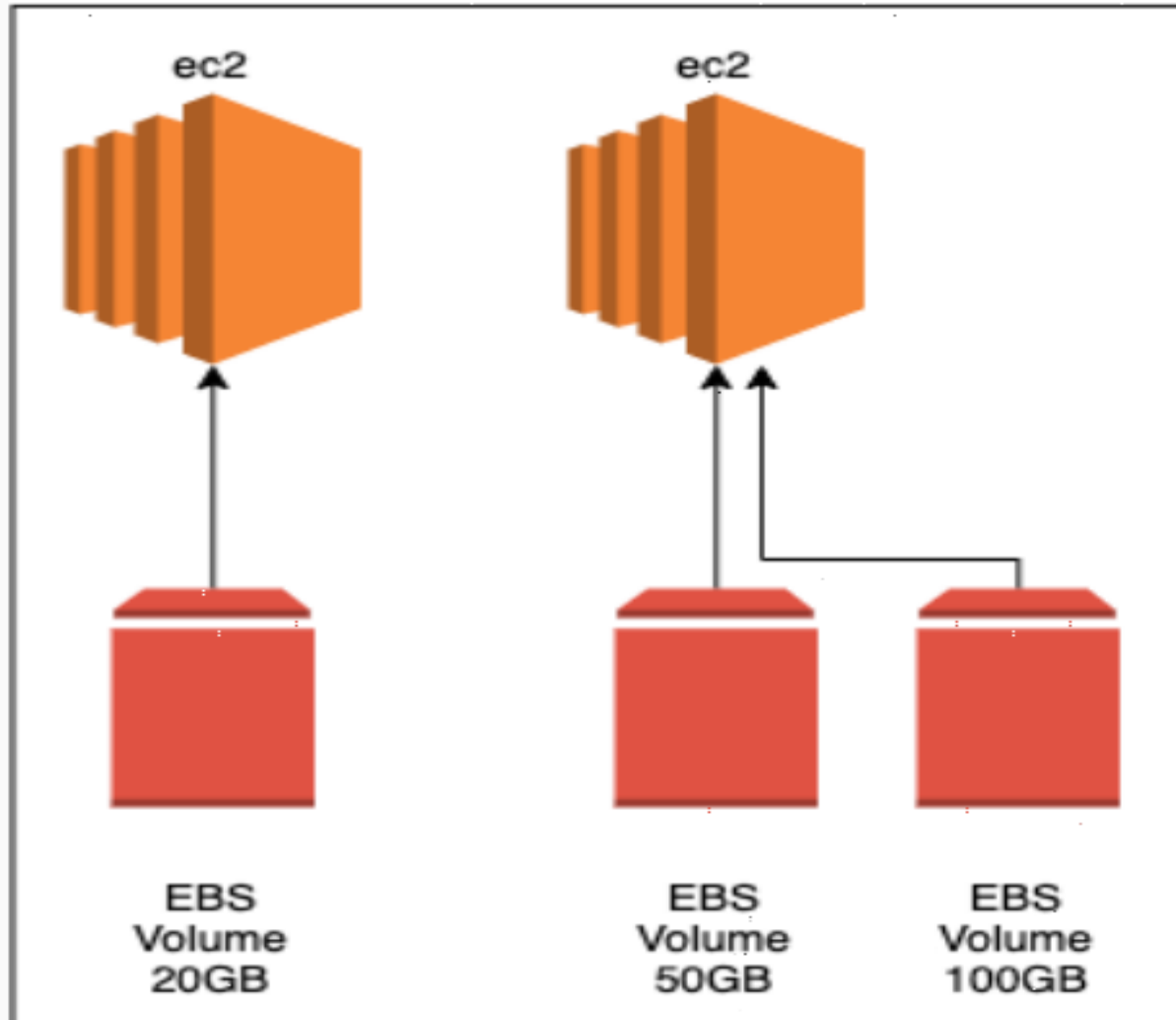
In traditional DC we have multiple dependencies on Hardware , Storage and OS teams. Hence a time consuming process.

With AWS Cloud its no more a pain, we can add, delete EBS volume on the fly.

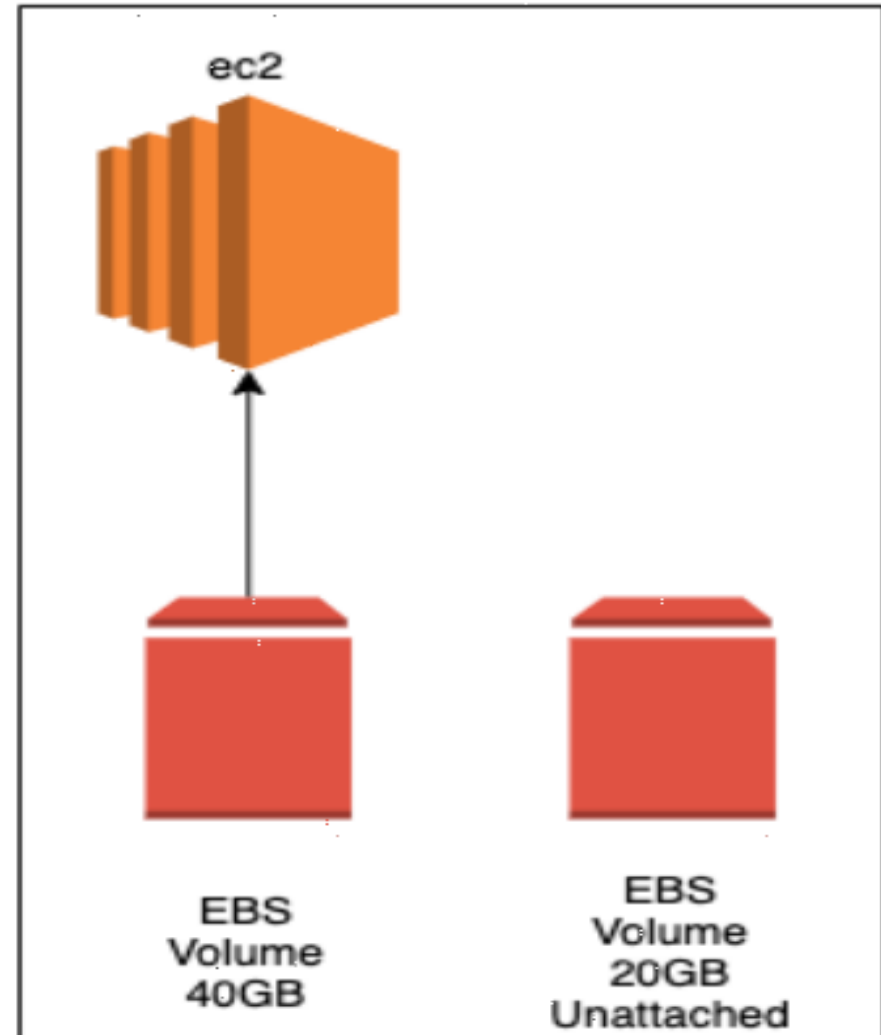
Once EBS volume is attached, you can reclaim the storage in few clicks and save the cost of storage by deleting them when not needed.



us-west-1a



us-west-1b



Amazon EBS Volumes

Amazon EBS

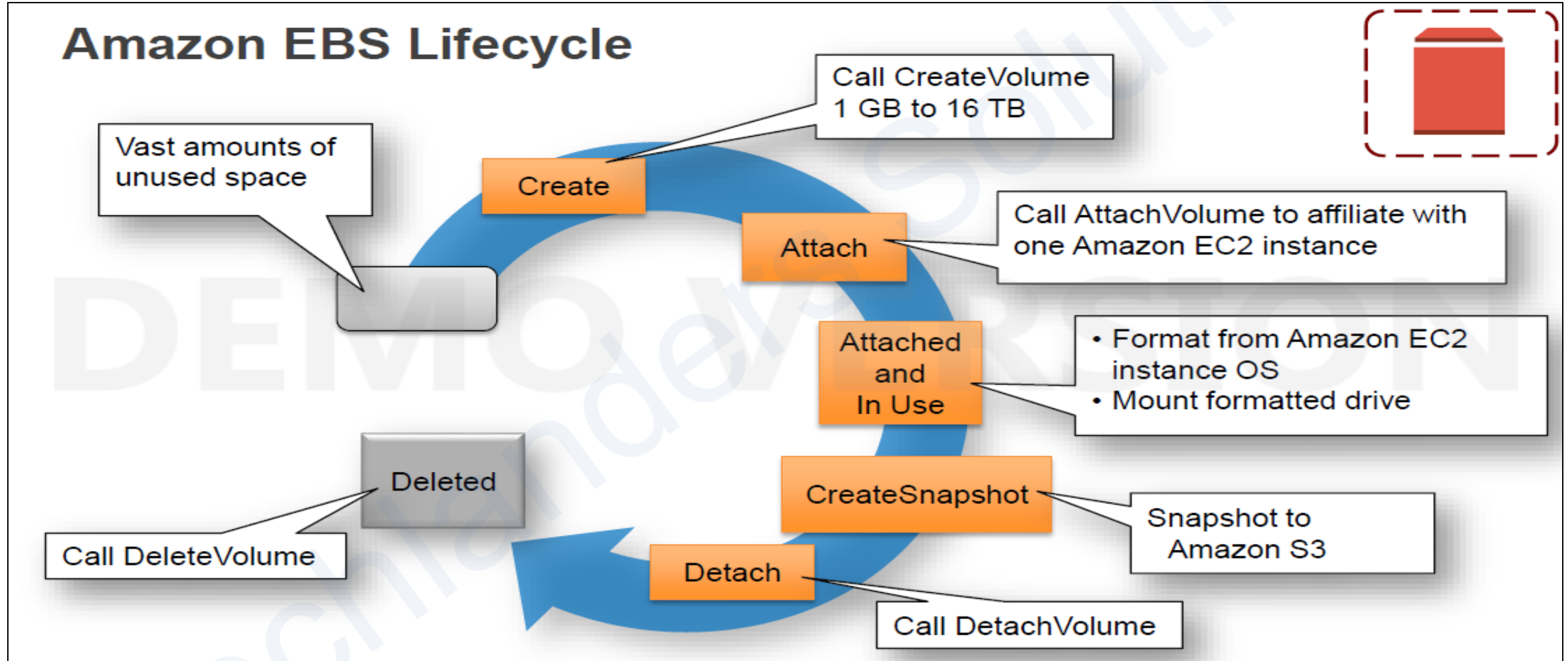
EBS stands for ***Elastic Block Storage***

Persistent block level storage volumes offering consistent and low-latency performance

Automatically replicated within its Availability Zone

Strictly bind to Availability Zone.

Amazon EBS Lifecycle



EBS Use Cases

OS – Use for boot/root volume, secondary volumes

Databases – Scales with your performance needs

Enterprise applications – Provides reliable block storage to run mission-critical applications

Business continuity – Minimize data loss and recovery time by regularly backing up using EBS Snapshots.

Applications – Install and persist any application



EBS Volume Type Comparison

	Solid-State Drives (SSD)		Hard disk Drives (HDD)	
Volume Type	General Purpose SSD (gp2)*	Provisioned IOPS SSD (io1)	Throughput Optimized HDD (st1)	Cold HDD (sc1)
Description	General purpose SSD volume that balances price and performance for a wide variety of workloads	Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads	Low cost HDD volume designed for frequently accessed, throughput-intensive workloads	Lowest cost HDD volume designed for less frequently accessed workloads
Use Cases	<ul style="list-style-type: none">• Recommended for most workloads• System boot volumes• Virtual desktops• Low-latency interactive apps• Development and test environments	<ul style="list-style-type: none">• Critical business applications that require sustained IOPS performance, or more than 10,000 IOPS or 160 MiB/s of throughput per volume• Large database workloads, such as:<ul style="list-style-type: none">◦ MongoDB◦ Cassandra◦ Microsoft SQL Server◦ MySQL◦ PostgreSQL◦ Oracle	<ul style="list-style-type: none">• Streaming workloads requiring consistent, fast throughput at a low price• Big data• Data warehouses• Log processing• Cannot be a boot volume	<ul style="list-style-type: none">• Throughput-oriented storage for large volumes of data that is infrequently accessed• Scenarios where the lowest storage cost is important• Cannot be a boot volume
API Name	gp2	io1	st1	sc1
Volume Size	1 GiB - 16 TiB	4 GiB - 16 TiB	500 GiB - 16 TiB	500 GiB - 16 TiB
Max. IOPS**/Volume	10,000	32,000***	500	250
Max. Throughput/Volume	160 MiB/s	500 MiB/s†	500 MiB/s	250 MiB/s
Max. IOPS/Instance	80,000	80,000	80,000	80,000
Max. Throughput/Instance††	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s	1,750 MiB/s
Dominant Performance Attribute	IOPS	IOPS	MiB/s	MiB/s

LAB 9 : Working with Volumes

Go to EC2 Instance created in -> EBS -> Volume -> Create Volume

Create a general purpose SSD Volume "Data-volume" of 1 GB

While creating volume, check different combination of IOPS and Throughputs, by selecting different-2 Volumes types

Attach volume to a server

Try to create another 1 gb volume but in a different AZ 2 and try to attach it with instance in AZ 1.

Detach volume from Server

Delete the volumes.

Make sure the volume and EC2 instance should be in same AZ to attach.